universität wien

DISSERTATION / DOCTORAL THESIS

Titel der Disseratation / Title of the Doctoral Thesis

„An Ontology-Based Cybersecurity Framework for the Automotive Domain - Design, Implementation, and Evaluation“

verfasst von / submitted by

Abdelkader Shaaban

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Doktor der Technischen Wissenschaften (Dr.techn.)

Wien, 2021 / Vienna, 2021

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	UA 786 880
Dissertationsgebiet lt. Studienblatt / field of study as it appears on the student record sheet:	Informatik
Betreut von / Supervisor:	ao. Univ.-Prof. tit. Univ.-Prof. Dipl.-Ing. Dr. Erich Schikuta Univ.-Prof. Dipl.-Ing. Dr. Dr. Gerald Quirchmayr

Acknowledgements

I would like to convey my deep sense of gratitude and my heartfelt appreciation towards all the personages who have assisted and supported me along the way of this dissertation. Without their cooperation, help, active guidance and encouragement, the work would not possibly come to the present shape.

First, I would like to sincerely extend my deep affection and appreciation to Prof. Mag. Dr. Abdel Baes Mohamed (AASTMT) who spared no effort supporting, encouraging and wisely guiding me throughout my research. I can not express enough thanks to him for enlightening my first glance of the research path since I started my Master degree under his supervision. His sense of paternity, vision, stimulating discussions and comments deeply encourage me.

I am ineffable grateful and indebted to my supervisors, Prof. Dr. Erich Schikuta and Prof. Dr. Dr. Gerald Quirchmayr, for their insightful guidance, support, and motivation. Their thoughtful feedback aimed to move me forward pushed me to sharpen my way of thinking and stepped my work to a higher level. They have taught me the methodology to carry my research out and present the work as clear as possible. I am tremendously privileged and honored to study and work under their guidance. Their invaluable expertise to inspired me to think outside the box, select the right direction and successfully accomplish this dissertation.

I am extremely thankful and grateful for MSc. Christoph Schmittner for sharing his immense knowledge, sincere collaboration, guidance, assistance and all the support I was given to further my research. It was a profit and pleasure to work with such a highly qualified person, to whom I convey all the respect and obligation. I sincerely wish him the best of luck and success in all his endeavours. I would like also gratefully thank Dr. Paul Smith for his precious time in proofreading the linguistic content of this dissertation. Also, my special thank goes to AIT - Austrian Institute of Technology for all possible support and opportunities I received to further my research and successfully complete this dissertation.

Finally, I would like to express my deep sense of reverence and gratitude towards my parents, who support me throughout my educational life emotionally and financially, they always never gave up for their positive belief in my success. Thank you both for giving me all opportunities and your knowledge to be who am I now. My deepest gratitude to my beloved and supportive wife for her patience, prayers understanding and continuous motivation. I would like to express my sincere love to my adorable little daughter Diana and my newborn baby Elina; I hope that I have made you proud of your father. I know I spent much time in my study and sorry for being busy and absent almost all the time.

Abstract

In view of the UNECE regulation for the approval of road vehicles, car manufacturers will have to ensure that their vehicles are cyber-secure. The increase of connected units in vehicles leads to a significant increase in attack surfaces, increasing the rate of potential security incidents. Therefore, security requirements verification and validation is a significant part of the development phase in current and future vehicles. We can only develop a secure vehicle if we define the existing security vulnerabilities that could be exploited by different potential threats and accurately select the relevant security requirements to address these security issues.

This work introduces an ontology-based security management framework for the automotive domain. The framework aims to check the correctness of the applied security requirements within the vehicular design by applying a series of logical inference rules to ensure that the security requirements are fulfilled. Sequences of procedures are applied for each vehicular component/asset individually to verify and validate the correctness of the selected security requirements. Additionally, it intends to manage the existing security gaps identified by the verification and validation process by suggesting a suitable set of security requirements that could be integrated within the vehicular design to protect the assets and the components within the vehicular network different forms of cyberattacks.

The framework is developed to be fully-adaptable for handling different forms of ontology inputs representing the relationships of the vehicular elements with existing security issues and the applied security requirements. This gives more flexibility to perform security requirements testing and manage the existing security issues in different forms. Four experiments are applied to a real case study in the automotive domain to examine the effectiveness of the proposed framework and demonstrate that the proposed security framework is adaptable to handle multiple input varieties. The outcomes prove that the framework is effective for saving time, effort, and reducing human mistakes. In addition, the framework provides a mapping strategy between different security requirements and the components/assets with their security vulnerabilities to focus on the applicable security requirements for addressing security issues. These requirements are considered a set of suggested or recommended security requirements, which assist in selecting and finding the most appropriate security requirements to fulfil security gaps in the vehicular design. This framework is suggested for the automotive sector. It could also be used and integrated within the development lifecycle of other relevant application domains such as in Cyber-Physical Systems and Internet-of-Things, in cases where the complete data are available. This will be discussed in further detail within this thesis context.

Kurzfassung

Im Hinblick auf die Regelung der UNECE für die Zulassung von Straßenfahrzeugen müssen Automobilhersteller in Zukunft sicherstellen dass ihr Fahrzeug cybersicher ist. Die Zunahme der vernetzten Einheiten in Fahrzeugen führt zu einer signifikanten Erhöhung der Angriffsflächen und damit zu einer Steigerung der Rate potenzieller Sicherheitsvorfälle. Daher ist die Überprüfung und Validierung der Sicherheitsanforderungen ein wesentlicher Bestandteil der Entwicklungsphase aktueller und zukünftiger Fahrzeuge. Wir können nur dann sichere Fahrzeuge entwickeln, wenn wir die vorhandenen Sicherheitsprobleme identifizieren und die passenden relevanten Sicherheitsziele definieren

In dieser Arbeit wird ein Ontologie-basiertes Sicherheitsmanagement-Framework für den Automobilbereich vorgestellt. Das Framework zielt darauf ab, die Korrektheit der angewandten Sicherheitsanforderungen innerhalb des Fahrzeugdesigns zu überprüfen, indem es eine Reihe von logischen Inferenzregeln anwendet, um sicherzustellen, dass die Sicherheitsanforderungen erfüllt sind. Eine Reihe von Prozeduren wird für jede Fahrzeugkomponente/jedes Gerät angewendet, um die Korrektheit der ausgewählten Sicherheitsanforderungen zu verifizieren und zu validieren. Darüber hinaus sollen die durch den Verifikations- und Validierungsprozess identifizierten Sicherheitslücken geschlossen werden, indem ein geeigneter Satz von Sicherheits-Maßnahmen vorgeschlagen wird, die in das Fahrzeugdesign integriert werden können, um die Assets und die Komponenten innerhalb des Fahrzeugnetzwerks vor verschiedenen Formen von Cyberangriffen zu schützen.

Das Framework wurde so entwickelt, dass es vollständig anpassbar ist, um verschiedene Formen von Ontologie-Inputs zu verarbeiten, die die Beziehungen der Fahrzeugelemente mit bestehenden Sicherheitsproblemen und den angewandten Sicherheitsanforderungen darstellen. Dies gibt Flexibilität bei der Prüfung der Sicherheitsanforderungen und der Verwaltung der vorhandenen Sicherheitsprobleme in verschiedenen Formen. Vier Experimente werden auf eine reale Fallstudie im Automobilbereich angewandt, um die Effektivität des vorgeschlagenen Frameworks zu untersuchen und zu demonstrieren, dass das vorgeschlagene Sicherheitsframework anpassungsfähig ist, um mehrere Eingabevarianten zu behandeln. Die Ergebnisse beweisen, dass das Framework effektiv Zeit und Aufwand spart und menschliche Fehler reduziert. Darüber hinaus bietet das Framework eine Zuordnungsstrategie zwischen verschiedenen Sicherheitsanforderungen und den Komponenten/Assets mit ihren Sicherheitsschwachstellen, um sich auf die anwendbaren Sicherheitsanforderungen für die Behandlung von Sicherheitsproblemen zu konzentrieren. Diese Anforderungen werden als eine Reihe von vorgeschlagenen oder empfohlenen Sicherheitsanforderungen betrachtet, die bei der Auswahl und Suche nach den am besten geeigneten Sicherheitsanforderungen unterstützen, um Sicherheitslücken im Fahrzeugdesign zu schließen. Das Framework wird für den Automobilsektor vorgestellt. Es kann auch in anderen relevanten Anwendungsbereichen wie Cyber-Physical Systems und Internet-of-Things eingesetzt und in den Entwicklungszyklus integriert werden, sofern vollständige Daten zur Verfügung stehen. Die Erweiterung auf weitere Domänen wird im Rahmen dieser Arbeit noch näher erläutert.

Keywords

Cybersecurity, Automotive, Potential Threats, Security Requirements, Ontology, Security Verification and Validation.

Contents

List of Tables

List of Figures

List of Algorithms

1 Introduction

The chapter gives an overview of the leading motivational background toward this research context. It describes the main research gaps in the automotive domain; in this work, we classify these gaps into practical gaps and a methodological gap. The main research contribution of this thesis is also included in this chapter. Then it addresses a set of research questions, which are going to be answered within this research work. The chapter ends with a summary of the thesis structure.

This chapter includes content from some articles published within this thesis work. The following list refers to the selected publications that are integrated within the context of the chapter:

- *Abdelkader Magdy, Shaaban, and Schmittner Christoph. 2020. "ThreatGet: New Approach Towards Automotive Security-By-Design." Pp. 413–19 in IDIMT-2020 Digitalized Economy, Society and Information Management. Kutná Hora, Czech Republic.*

- *Shaaban, Abdelkader Magdy, Christoph Schmittner, Thomas Gruber, A. Baith Mohamed, Gerald Quirchmayr, and Erich Schikuta. 2019. "Ontology-Based Model for Automotive Security Verification and Validation." Pp. 73–82 in Proceedings of the 21st International Conference on Information Integration and Web-Based Applications &; Services, iiWAS2019. New York, NY, USA: Association for Computing Machinery*

- *Shaaban, Abdelkader Magdy, Christoph Schmittner, Thomas Gruber, A. Baith Mohamed, Gerald Quirchmayr, And Erich Schikuta. "Automated Ontology-Based Security Requirements Identification for The Vehicular Domain." Journal of Data Intelligence 1, No. 4 (2020): 401-418.*

- *Shaaban, Abdelkader Magdy, Christoph Schmittner, Gerald Quirchmayr, A. Baith Mohamed, Thomas Gruber, and Erich Schikuta. 2019. "Toward the Ontology-Based Security Verification and Validation Model for the Vehicular Domain." Pp. 521–529 in Neural Information Processing, edited by T. Gedeon, K. W. Wong, and M. Lee. Cham: Springer International Publishing.*

1.1 Research Motivation

Cybersecurity is one of the essential parts of any system engineering development lifecycle. It is responsible for protecting data from different cyberattacks such as integrity violation, information disclosure, spoofing, and many other forms of malicious activities. In the automotive domain, cybersecurity is responsible for protecting the vehicular data and protecting the critical vehicular units from different cyberattacks. Modern cars consist of a wide range of electronic and electrical units necessary for a safe and comfortable driving. The attacker could compromise a critical vehicular unit such as the engine control unit or brake control unit, leading to unexpected negative consequences ranging from injuries to death.

The United Nations Economic Commission for Europe (UNECE) World Forum for Harmonization of Vehicle Regulations (WP.29) provides a framework for harmonized vehicle regulation [UNE20c]. WP.29 is a unique regulatory mechanism within the UNECE Inland Transport Committee (ITC) framework. It enables the market to present innovative technologies for vehicles with a focus on enhancing vehicle safety. Also, it intends to minimize environmental pollution and energy consumption. In addition, the regulation promotes cross-border trade since it is established under the 1958 agreement. It also includes approval of the vehicle systems, components, and equipment [UNE20d]. The type approval describes the process that is used by authorities to verify that a particular vehicle meets all EU regulations, such as safety and environmental specifications, before being placed on the EU market. The vehicle manufacturer makes prototypes similar to the final product used to examine compliance with safety rules, emissions, and manufacturing requirements such as seats or steering wheel airbags. If all relevant specifications are satisfied, the national authority grants EU vehicle type approval certification to the manufacturer to authorize the EU vehicle type sale. Conformity certification confirms each vehicle and can be thought of as a vehicle's birth certificate, showing that the vehicle matches an approved type [Com16]. The 1958 agreement has expanded to 54 Contracting Parties (CP), including all EU nations and others such as Australia, Japan, Russia, South Africa, and Turkey. Figure 1.1 illustrates the nations, which are participated in the 1958 agreement [Sec20].

Figure 1.1: The 54 contracting parties in the 1958 agreement [UNE19]

The primary duty of the WP.29 is to keep the vehicle regulations up-to-date and appropriate, particularly in terms of technology, safety and environmental change [Sec20]. In June 2018, WP.29 changed the Brakes and Running Gear (GRRF) working party into a new Automated/Autonomous

and Connected Vehicles (GRVA) working party [UNE20b]. Proposal for a new UN Regulation has been released regarding the approval of vehicles cybersecurity and cybersecurity management system (CSMS) that is drafted by the Task Force on cybersecurity and over-the-air issues (TF-CS/OTA) and then reviewed by GRVA. The TF-CS/OTA develops two recommendations; the first is a recommendation on cybersecurity, where the second is to develop a recommendation on the software updates. Figure 1.2 illustrates the task force activities.

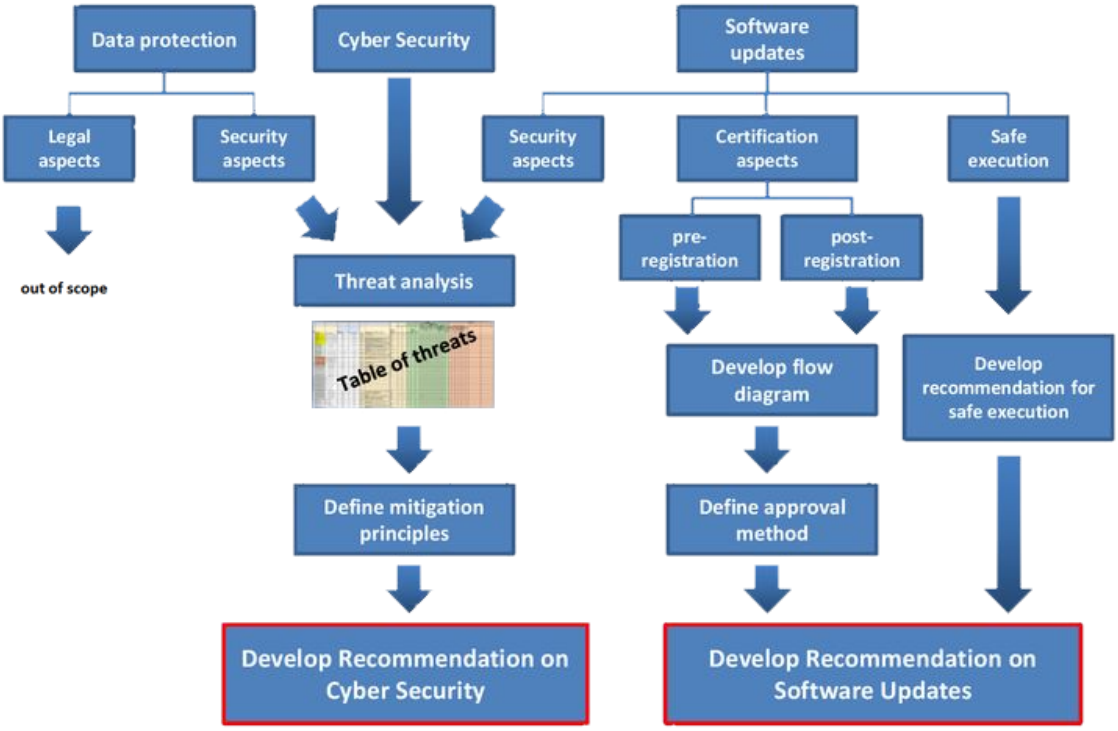


Figure 1.2: The recommendations of the task force on cybersecurity and over-the-air issues [UNE18]

The vehicle manufacturer shall prove that the CSMS applies to the development process including production and post-production phases. Annex 5 in this draft version includes a set of threats and mitigations [Nat20]. These threats and the related impacts are used in this thesis work as a set of potential threats related to the automotive domain for performing testing on security requirements to the related potential threats.

In order to go more deeply into automotive cybersecurity, we need to know first what is automotive cybersecurity?. According to NHTSA [1], Automotive Cybersecurity is:

" *Cybersecurity, within the context of road vehicles, is the protection of automotive electronic systems, communication networks, control algorithms, software, users, and underlying data from malicious attacks, damage, unauthorized access, or manipulation*" [NHT16].

The vehicular industry has seen significant development in automotive design, in which vehicular units are evolved from mechanical parts working on gears and shafts to electronic/electronic

[1] https://www.nhtsa.gov

components communicating through communication protocols. Modern vehicles combine a considerable number of interconnected units such as Sensors, Electronic Control Units (ECUs), Buses, Actuators, and other electronic elements for monitoring and controlling the state of the vehicle [MV13]. Next-generation vehicles will include 20+ computers with storage sizes ranging from 8GB to 256GB [MUT17]. The high-end car has over 100 million lines of code, and it is expected that this number will continue to grow. Such codes are implemented for various control applications over numerous functionalities like safety-critical functions, driver-assistance, and others. The software operates on hundreds of programmable ECUs that interact via several types of communication protocols and buses (i.e., Controller Area Network (CAN bus), FlexRay, and Ethernet) [CAFC+16].

In the future, the manufacturers of vehicles will need to check the cybersecurity of their vehicle systems before their automobiles can be accepted for sale. Every three years, they have to demonstrate that they have implemented a certified cybersecurity management program for covering all processes, from vehicle engineering to documentation [AIT19]. Cybersecurity plays a vital role in the automotive domain because it protects critical vehicular units and software, which controls functional safety in a vehicle from various attack scenarios. Accordingly, the safety-security relationship is considered directly proportional to any malicious code injected into the vehicular network, that leads to damage or failure of either a particular unit or the whole vehicular system [SSG+19]. New automotive systems will need to ensure safety and security; where security analysis is considered a crucial factor in this process [MS16].

In the vehicular industry, safety concerns are increased regarding the replacement of mechanical units with embedded systems [MABK16]. The integration of the internet connectivity with modern vehicles to provide advanced features such as autonomous driving, communicating with cars or other external nodes, software updating, and others, opens a new relevant topic in the vehicular industry which is cybersecurity. Any device connected to the Internet can be exposed to multiple kinds of malicious attacks; the same is true for the modern vehicle [McA16]. A cyber-secure vulnerable point in a car could lead to the entire vehicle being out of control. An attack could happen in case malicious code is injected into communication buses in a vehicle that could affect the standard functionality of the vehicle and cause tragic consequences. Moreover, it is necessary to study the exact security vulnerabilities in the initial phases of the vehicle engineering process because afterwards, it becomes cost-ineffective and challenging to add security countermeasures [KN17].

Another critical issue in automotive cybersecurity is who controls access to data that is collected by the vehicle, as discussed in [SLSH18]? According to [oPF17], there are multiple data categories collected by the vehicle. The following list discusses some of the most common data that are collected by the vehicle:

- **Event Data Recorders - EDR:** record information about the operation of a vehicle. This information is collected to help in the investigation of an accident or crash. It could be information about speed, accelerator and brake position, the usage of seatbelts, and other critical information.
- **On-Board Diagnostic Information:** This information is collected by connecting a suitable device throughout the On-Board Diagnostic port - OBD port. This port enables access to the vehicular information for diagnostic purposes. The car maintenance service could collect this information to assist in diagnosing a particular failure or an issue in a vehicle.
- **External Information:** Modern cars contain sensors and cameras for collecting information

about the surrounding environment. The sensors could collect some external information, such as weather conditions, obstacles, lane indicating, or others.

- **Apps:** pairing smartphones with the vehicle becomes possible with a lot of advanced features. The vehicle may allow an interface between the connected devices (i.e., phones, tablets, etc.) and the vehicle itself. Enabling this service could allow a third-party to get access to your data through the car.

Based on the above examples of common data collected by the vehicle, we can say that any existing security vulnerabilities within the vehicle network could affect any of these data. Multiple malicious ways could affect these data, such as information disclosure, tampering, spoofing, and many other malicious activities. As presented in [SLSH18], personal information is exchanged in a car without encryption. This means that when a vehicle is under attack, a user's critical personal data such as credit card information, passwords, images, contacts number, and other critical data are in danger.

Security becomes a critical issue in the automotive domain. It is responsible for protecting the user's sensitive personal data and protecting critical vehicle units that control the vehicle's functional safety from different types of malicious activities that could lead to unexpected negative consequences.

1.2 Research Gap

This section classifies the primary research gaps into two main categories. The first category is based on practical security issues in the automotive domain, where the second category is the methodological gap:

- **Practical problems necessitating research:**
  - **P1:** Regarding the regulation of the UNECE WP29, connected/autonomous cars will require future vehicles to be much more cyber secure. In the future, automotive manufacturers will need to check how secure their cars are, as discussed in [AIT19] [UNE20a].
  - **P2:** The increase of connected units in vehicles leads to a considerable number of attack surfaces, which possibly leads to an increasing amount of security incidents. In this case, the process of security verification, and validation (V&V) will be more complicated because this process needs to be aware of all vehicular components, assets, potential threats, and related security requirements, which is considered a challenging process. According to the UN Regulation [UNE20a], the vehicular manufacture shall ensure that the aspects of cybersecurity defined in the regulation are implemented.
  - **P3:** The ISO/SAE 21434 security standard describes the security engineering process in the automotive domain and intends to secure the whole life cycle in the vehicle development stages. The first draft of the standard was published in February 2020. However, the final standard is expected to be published in early 2021, as mentioned in [Til20]. Regarding the absence of automotive security requirements, it opens the way to select security requirements from other standards to tackle and close this gap. However, this is a challenging process because it needs to select security requirements to solve associated security issues, which could be a time-consuming process and could not guarantee security compliance with a specific security standard.

- **Methodological Gap:**
  - **M:** The practical gaps motivate the need for a methodological framework to automate the mapping between security weaknesses and related security requirements in the vehicular domain. This approach studies the relationships between all the vehicular entities (i.e., components, assets, security measures, threats, security requirements, security levels, and severity degrees) to define the applicable security requirements to fill the gap in the vehicle design. Most of the mapping approaches are based on manual activities. Furthermore, by automating this approach, we can address the above practical gaps by increasing the accuracy of the V&V, reducing human mistakes, saving time and cost, and providing a complete mapping strategy between different security requirements from alternative resources. Also, it suggests the most applicable security requirements for addressing potential existing threats according to UN regulation.

    For example, the protection profile describes the security considerations and resulting requirements for a Target of Evaluation (ToE) according to Common Criteria (CC) [ISO09b]. The ToE is an abstract description of a system or a system unit for specific usage. The PP identifies Security Target (ST) or security properties of ToE(s). It is essential to ensure the compliance of one or more PP(s) with identified ToE(s) to develop secure vehicles. This is especially important since systems designed for vehicular usage are often reused in a different context. Assuring that such a system complies with the PP for this context ensures that the security needs are covered. According to the [noa17a], the security requirements rationale that explains why this particular set of Security Assurance Requirements - SARs (i.e., describe the evaluation of theTOE) was considered as suitable. However, there is no particular explanation for that. Therefore, design such as this methodological approach could help in filling the above discussed practical gaps.

1.3 Research Questions

In order to tackle the main discussed security gaps, the need for a cybersecurity framework is required. Therefore, a set of questions shall be raised to clearly understand the possible solutions that could be proposed to solve such security issues in the automotive domain. For this purpose, we outline three main questions that can formalize the main contribution of this dissertation; these questions are defined as follows:

- ***RQ1. How can an ontology-based cybersecurity framework for the automotive domain be built?***
  - ***RQ1.1. How can security vulnerabilities in the vehicular domain be treated with a focus on the interaction between the internal and external units in the vehicular systems?***
  - ***RQ1.2. How can an appropriate ontology model support in the automotive cybersecurity be built?***

  The ontology structure helps find common security characteristics between vehicular components/assets, threats, and security requirements. The cybersecurity framework establishes multiple classes, categories, entities, properties, and annotations for all the required details

(i.e., threats, security requirements, security properties, components, assets, risk severity, and others). The framework is considered as a metamodel-based structure. The metamodel describes the structure of modelling phases [TSL+06]. Therefore, the metamodel is conducted in this research to describe the ontology's main structure for building a complete overview of entities and relationships. This structure represents the ontology taxonomy of the automotive cybersecurity, which is used to develop the ontology-based framework. **Section 3.6** describes more details about the structure of the ontology model.

The cybersecurity framework applies a set of inference rules to deduce the most applicable security requirements according to the exiting security vulnerabilities to address the identified potential threats and protect the vehicular units from different cyberattacks. Therefore to build and design a secure vehicle infrastructure, we need to:

- identify the exact potential threats,
- selects an applicable set of security requirements accurately.

Chapters 3, 4, and 5 give more details in answering this research question.

- ***RQ2. What are the main activities that should be integrated within this work to implement a reliable cybersecurity framework in the vehicular domain?***

  The proposed framework consists of five main phases (i.e., data digestion, verification, validation, gap analysis, and security enhancement). Each phase has a particular activity for evaluating the correctness of the security requirements and handling the existing security gaps by an applicable set of security requirements. **Chapter 6** gives more insight into the main structure of the proposed framework and discusses more details about the main activities for each phase of the cybersecurity framework.

- ***RQ3. What is the main research impact on the automotive domain?***
  - ***RQ3.1 What are the main impacts of the proposed framework on the cybersecurity engineering process in the automotive domain?***
  - ***RQ3.2 How can the effectiveness of the cybersecurity framework in the automotive domain be demonstrated?***

The proposed framework is designed, developed, and implemented to be fully-adaptable for various kinds of ontology inputs model that represents the interactions between the different components/assets in the vehicle network. It handles different types of input forms of the vehicular ontology representation model, such as follows:

- Integration of both threats and security requirements.
- Import of potential threats.
- Import of security requirements.
- No remaining threats or uncovered security requirements.

Different experimental outcomes are defined according to these types of input forms based on the structure of the main case study that is used in this research context. **Chapter 7** gives a complete view of these experiments and demonstrates the effectiveness of the cybersecurity framework for ensuring the correctness of the applied security requirements of the vehicular design and demonstrates its ability for handling the existing security issues.

1.4 Research Contribution

Throughout five years of my PhD work, different cybersecurity approaches conducted and introduced to resolve existing security issues in the automotive domain and other relevant research topics such as the Internet of Things — IoT, Cyber-Physical Systems — CPS, Cyber-Physical Production Systems — CPPS, and Railways. As part of this research context, several publications were published to demonstrate our efforts within the research community and share the latest outcomes to prove the correctness of our theories. These publications are stated and listed in the early part of this dissertation context, with a brief discussion in the "Publications" section.

Threat analysis and security requirements management are the main research direction that I have conducted and focused on throughout my PhD research. Threat analysis helps determine a transparent risk investigation approach for focusing on the security issues threatening different application domains. In order to build secure systems, we need to define a set of applicable security requirements that need to be a part of the system design to protect a critical system's assets from different malicious activities. Therefore, the investigation of defining the applicable security requirements is also an essential part of this research work. The Ontology approach is integrated within this research to manage a wide range of potential threats and maintain the diversity content of security requirements are collected from multiple resources. The ontology helps in creating a comprehensive semantic knowledgebase of data used within this research context.

As discussed in the research motivation **Section 1.1**, cybersecurity is considered one of the critical issues in the automotive industry. On the first hand, it is responsible for protecting the vehicular data and personal data from cyberattacks. On the other hand, it also protects the critical vehicular units that control and manage the functional safety in the vehicle from different malicious actions which lead to unexpected negative consequences. The key research goal is to address the previously discussed research gaps in automotive cybersecurity, as discussed in **Section 1.2**. The methodological framework that can track, understand, and create relationships between security vulnerabilities and related security requirements in the automotive domain is needed to tackle the previously discussed research gaps, which is considered the primary motivational methodological point for this PhD's contribution.

Furthermore, this research designs, implements, and evaluates an ontology-based cybersecurity framework for the automotive domain. The framework examines the correctness of the applied security requirements within the vehicle design for protecting the vehicle from different cyberattacks. It also manages and handles the existing security gaps in the vehicular design by focusing on the most appropriate security requirements. These security requirements shall contain common security properties with threats and particular components/assets. The framework provides a proposed concept of managing security issues in the vehicular domain by providing complete automation process to save time and effort that a highly-experienced vehicular security architect spends. The proposed framework is intended to be integrated at the early stages of the security engineering process for addressing the existing security issues. This is necessary because once the vehicle is built, it becomes more difficult to add security.

Protection profile (PP) could be a way of defining security requirements per security objectives. These requirements are defined as groups for each security objectives. There has to be at least one objective for every security requirement, as discussed in [Mar20]. PP represents a **Security Objectives Rationale** (i.e., $Threats \longrightarrow$SecurityObjectives), which defines a tracking between threats and security objectives. Also, it defines **Security Requirements Rationale** (i.e., $SecurityRequirements \longrightarrow$SecurityObjectives) to represent a tracking of security requirements that are chosen to reach security objectives. However, it is unclear which methodological approach

should be followed to track threats and relevant security requirements to achieve the main security objectives.

The proposed cybersecurity framework tackles this issue, which automates tracking threats and related security properties until security requirements are defined to achieve a particular ST. It creates relationships among multiple entities in the hierarchical ontology model based on facts and hypotheses (as discussed in Chapter 3) to deduce security requirements for addressing threats and protecting particular vehicular components/assets. These relationships provide an in-depth explanation about why a set of security requirements are selected for a particular security issue. These relationships are built according to security properties are extracted from threats, components/assets and security requirements to estimate existing relationships and create new ones, as follows:

- *Components/Asset* $\longrightarrow$SecurityProperties "A component/asset has security properties to protect it"
- *SecurityProperties* $\longrightarrow$Threat "Security properties could be vulnerable points that an attacker can exploit by multiple ways, which are defined as threats"
- *Threat* $\longrightarrow$Components/Asset "A threat impacts a component/asset"
- *Threat* $\longrightarrow$*SeverityLevel* "Each threat has a degree of risk level"
- *SeverityLevel* $\longrightarrow$*SecurityLevel* "Severity level has an equivalent security level that handles security issues"
- *SecurityLevel* $\longrightarrow$*SecurityRequirements* "Security requirements are classified according to multiple security levels"
- *SecurityRequiremets* $\longrightarrow$SecurityProperties "A set of security properties can identify security requirements to describe security measures applied to achieve particular security objectives"
- *SecurityRequirements* $\longrightarrow$Component/Asset "According to that, security requirements are chosen to address a threat and protect component/asset"
- *SecurityRequirements* $\longrightarrow$Threats "Security requirements are chosen to address a threat"

Therefore, the framework defines security requirements according to facts and hypotheses for creating a set of relationships to define a direct tracing methodology from threats to security requirements. These relationships build a complete tracking path from threats to security requirements for addressing security vulnerabilities in the vehicular design. They are also assisted in enhancing outcomes of the security verification process (theorem proving, as discussed in **Section 6.3**) to check the logical correctness of the selected security requirements. In addition, build reverse engineering methodology to validate the correctness of the applied security requirements against existing and expected threats, as discussed in **Section 6.4**. The next chapters emphasise this methodology and demonstrate the effectiveness of the proposed framework in a real case study.

The essential part of this work is building a complete knowledge representation of the vehicle architecture model by defining all relevant details involved within the vehicular network. The ontology approach helps to define all the details needed to define the relationships among the

different components, assets, threats, security requirements, security measures, risk levels, and severity degrees in the vehicular model. The ontology is used to create a complete taxonomy model of the vehicular design with all the details about the components with the relationships of existing security issues and relevant security requirements, **Section 3.5** and **Section 3.6** give more details about the structure of the proposed ontology design. The security requirements integrated with this work are based on alternative data resources. It is necessary to select reliable data with a high level of genuineness to prove the efficiency of the work. Therefore, this research includes IEC 62443 part 4-2 security standard and security requirements from the V2X Hardware Security Module Protection Profile according to the common criteria. **Chapter 4** gives a complete view of these data, their structure, and how these data are integrated into this research work. Likewise, in selecting the potential threats, several resources are examined to decide the most appropriate resources to be integrated within this research context. Therefore, we use the UNECE threats list, which is a set of potential high-level threats and relevant security vulnerabilities in the automotive domain. Also, the threats defined in the V2X Hardware Security Module Protection Profile are used and integrated within this work. According to the design of the case study as discussed in **Chapter 7**, where we use the Common Vulnerabilities and Exposures - CVE to choose a set of threats that could be used within this research work. **Chapter 5** provides more information on the threats, resources, and approach to translate threats into an ontology representation to be incorporated into this work.

The framework is developed to be fully-adaptable to handle different types of ontology input forms that represent the vehicular data representation. The proposed cybersecurity framework consists of five main phases (i.e., data digestion, verification, validation, gap analysis, and security enhancement); these phases are discussed, and explained in **Chapter 6**. **Section 6.2** explains the activity of the digestion phase, which reads and scans all input data in the vehicular ontology form. Then the framework verifies the security characteristics between the different individuals within the vehicle design. The verification is based on the theorem proving approach for logically ensuring the correctness of the interactions among the individuals (i.e., components, assets, security properties, severity degrees, and security levels) of the vehicular ontology model. The verification phase is discussed in detail in **Section 6.3**. The testbed execution is applied to perform a set of sequence procedures for each vehicular component/asset individually to validate the correctness of the applied security requirements into the vehicular design. This phase consists of a set of processes that are inspired by the ISO risk management process (i.e., risk identification, analysis, evaluation, and treatment). The framework applies a set of rules to the ontology form to determine the correctness of the applied security requirements based on risk management activities. These activities examine each vehicular component/asset separately and define the potential threats that could have an impact on the selected vehicular unit.

Then the framework performs the risk analysis to determine the exact security threats that are defined according to the applied security properties on the vehicular component/asset. The risk evaluation process evaluates the outcomes to estimate the security target of the vehicle component/asset. The framework deduces an applicable set of security requirements for addressing the identified threats. The outcomes are compared with the existing data (i.e., threats and security requirements) to determine the standardization of the applied security requirements against the potential exiting threats for a particular vehicular component/asset. The validation phase also performs a reverse action from the existing security solution that infers new security issues to investigate the effectiveness of the applied security requirements against the exact potential threats. This process aims to ensure that the selected security requirements are out of redundancy or ineffectiveness against the existing security issues. **Section 6.4** explains more details about the

main activities and processes developed within the validation process.

The gap analysis and security enhancement phases, as discussed in **Section 6.5**, aim to handle and manage the existing security gaps that are discovered in the verification and validation phases. Therefore, the framework also aims to address the existing security gaps to protect the vehicular assets/components from different malicious actions. Additionally, it manages these gaps to improve the overall security level to meet the actual security target. The inferred security requirements in the validation phase are used to fill the current security gaps in the vehicular design. These requirements are considered a set of suggested or recommended security requirements, which assist the system designer in selecting and finding the most appropriate security requirements that help to fulfil the existing security gaps in the vehicular design. The framework gives initial estimations of the security target, and the security achieved level after the suggested security requirements are applied. These estimations are assumed as preliminary numbers to define the first indication of the achieved security level against the required security target level, which indicates how far the security target to be achieved.

Figure 1.3 summarizes the main contribution of this work into distinct processes using Business Process Model and Notation - BPMN.

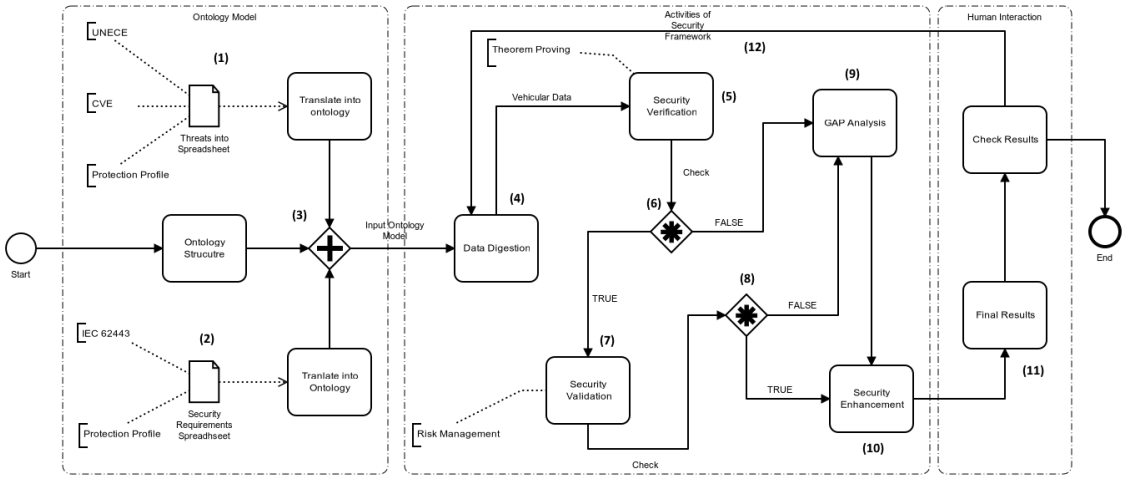


Figure 1.3: The process model of the research contribution

The following points clarify the main steps that the cybersecurity framework follows for achieving its goal:

(**1**): Potential threats and vehicular elements are collected and translated into ontology representation form.

(**2**): We extract data from threat resources (UNECE, CVE, and protection priles) used in this work and create a spreadsheet with all the required data. Similarly, in the security requirements process, the collected data (i.e., IEC 62443 and protection profile - common criteria), are defined in a spreadsheet.

(**3**): Then threats, vehicular elements and security requirements ae combined and translated into the ontology representation model according to the main structure of the ontology. The ontology representation model is thought of as an input to the proposed cybersecurity framework.

(**4**): The data digestion activity aims to read and scan all input data in ontology form.

(**5**): The verification activity checks almost every node in the ontology design to verify the logical correctness of all vehicle entities in the ontological model.

(**6**): Check results to detect any existing security gaps.

(**7**): The validation process checks the correctness of the applied security requirements against potential threats. The validation is based on a testbed approach consists of processes according to the ISO's risk management process (i.e., risk identification, analysis, evaluation, and treatment).

(**8**): The unsatisfied results are defined as security gaps in the vehicular system design.

(**9**): Security gaps are handled by security enhancement activity for addressing these security issues.

(**10**): The framework suggests security requirements that could be integrated with the vehicular design for improving the overall security target level.

(**11**): The final results are collected by the vehicular security architect for review and evaluation.

(**12**): The framework could receive new ontology input model and repeat the whole process starting from step (4).

1.5 Thesis Structure

This section gives an overview of the contents of this thesis. This thesis contains nine chapters that are defined to describe the main concept of this research work.

Chapter 1: Introduction: This chapter gives an introductory discussion for this research endeavour. **Section 1.1**, discusses the main research motivation points to identify the primary purpose of this research work. **Section 1.2** represents the main research gaps in automotive cybersecurity, which are going to be addressed within this research context. The research questions are discussed briefly in **Section 1.3**, which describe the main topics that are going to be explained within this dissertation. The main contribution of this dissertation is discussed in **Section 1.4**, which presents an overall discussion about the main contribution of this research to address the previously discussed research gaps.

Chapter 2: State of the Art and Research Context: This chapter gives an overview of the state-of-the-art in automotive cybersecurity. **Section 2.1** presents different levels of automation in modern vehicles and introduces a short overview of the safety background in the automotive sector. The section also discusses the importance of cybersecurity in protecting critical vehicular units, and how cybersecurity in the automotive domain could assist in reducing accident rates. Cybersecurity challenges are discussed in **Section 2.2**. This section includes a study on different research approaches that are conducted in the automotive area. Different threat modelling methods are discussed in **2.2.1**, this section addresses how the threat modelling can be used in the automotive cybersecurity investigation. **Section 2.2.3**, discusses how to protect the vehicle from different cyber attacks and fill the gaps of the existing security vulnerabilities by selecting the proper security

requirements. An overview of the existing security standards from relevant domains is discussed in **Section 2.2.4**. The mapping between ontologies is discussed in **Section 2.3**. Ontology in cybersecurity is also presented in **Section 2.4**.

Chapter 3: Research Questions and Methodology Approaches: Chapter 3 gives an introduction to the research methodology followed in this context to address the key research problem. The research questions are discussed in **Section 3.1**; this section aims to formalize the main contribution research details into a set of questions to improve the overall understanding of the leading dissertation topic. The research problem and relevant facts, hypotheses, theory and strategy, are discussed in **Section 3.2**. The research Justification/Evaluation and Knowledge-Based are considered in **Section 3.3** and **Section 3.4**, respectively. **Section 3.5** explains the structure of the proposed ontology model that is developed and integrated within this work. The main structure of the classes, sub-classes, individuals, data objectives, and data properties of the ontology structure is presented in **Section 3.6**. **Section 3.7** introduces the verification and validation process that is conducted in this research.

Chapter 4: Research Data Collection: Security Requirements: This chapter gives more details on the security requirements resources that are used in this work, as discussed in **Section 3.4.1**. In this work, we use two primary resources for building our security requirements knowledge-based. **Section 4.1** describes the contents of the resources we use within this research work. The primary data extracted from these resources are discussed in **Section 4.2**. The extracted data is presented in a spreadsheet; then, it is translated into an ontological form. **Section 4.3** discusses how the extracted security requirements data could be translated into ontology form.

Chapter 5: Research Data Collection: Potential Threats and Vehicular Components/Assets: This chapter presents further details on the potential threats and the classification of the vehicular components that are used in this research, as discussed in **Section 3.4.1**. In this work, we use three essential resources for creating our potential threats knowledge-base. **Section 5.1** represents the contents of the resources we use within this research work. The primary data extracted from these resources are discussed in **Section 5.2** and **Section 5.3**. The extracted data is presented in a spreadsheet; then, it is translated into an ontological form. **Section 5.4** discusses how the extracted potential threats and components/assets data contents could be translated into ontology form.

Chapter 6: An Ontology-Based cybersecurity Framework: This chapter highlights the fundamental structure of the proposed cybersecurity framework. **Section 6.1** discusses the main structure of the proposed cybersecurity framework. Understanding and scanning the received data input is discussed in **Section 6.2**. **Section 6.3** explains that the main verification approaches are applied in this research. The testbed execution method for the validation process is presented in details in **Section 6.4**. This section includes a set of processes that are followed by the proposed framework for performing the validation activity for each vehicular component/asset. These activities are based on ISO 27005, according to ISO 31000. The risk identification process is presented in **Section 6.4.1**. **Section 6.4.2** explains the main activities are involved in this process. Evaluating the identified risks are discussed in **Section 6.4.3**. Then risk treatment is presented in **Section 6.4.5**. The gap analysis and security enhancement processes are explained in **Section 6.5**,

Chapter 7: Use Cases and Experimental Model Evaluation: This chapter introduces an example of a real use case in the automotive domain to evaluate the reliability and validity of the proposed cybersecurity framework. **Section 7.1** discusses the structure of the case study that is used within this research context. Four different experiments are created to describe different input forms for the case study that can be handled and managed by the framework; these four examples are shown in **Section 7.2**. **Section 7.3** presents the outcomes of all the four created experiments, and also evaluates the results for each one. A summary of the overall evaluation and comparison between the framework and the manual approach is addressed in **Section 7.4**.

Chapter 8: Summary and Conclusion: This chapter summarizes the whole thesis work, which provides an overview of this work's research contribution and research impact as discussed in **Section 8.1** and **Section 8.2** respectively. **Section 8.3** discusses the main research limitations and provides recommendations as new ideas to cope with the existing limitations.

Chapter 9: Thesis Statement: This chapter describes the main research points that represent the actual research contributions and outcomes.

2 State of the Art and Research Context

This chapter introduces the state of the art of this research and presents the relevant work of this thesis. The chapter starts by giving an overview of modern vehicles. It discusses the different automation levels starting from no automation, and how that evolved from mechanical parts to complete cyber-physical systems. Then the chapter includes a discussion about safety engineering in modern vehicles. The cybersecurity in the automotive domain is considered afterwards, to give an in-depth discussion on cybersecurity in modern automobiles and how security vulnerabilities in the vehicle design could lead to many unwanted consequences.

This chapter includes content from some articles published within this thesis work. The following list refers to the selected publications that are integrated within the context of the chapter:

- *Abdelkader Magdy, Shaaban, and Schmittner Christoph. 2020. "ThreatGet: New Approach Towards Automotive Security-By-Design." Pp. 413–19 in IDIMT-2020 Digitalized Economy, Society and Information Management. Kutná Hora, Czech Republic*
- *Shaaban, Abdelkader Magdy, Erwin Kristen, and Christoph Schmittner. 2018. "Application of IEC 62443 for IoT Components." Pp. 214–23 in Computer Safety, Reliability, and Security, Lecture Notes in Computer Science, edited by B. Gallina, A. Skavhaug, E. Schoitsch, and F. Bitsch. Cham: Springer International Publishing.*
- *Shaaban, Abdelkader Magdy, Christoph Schmittner, Thomas Gruber, A. Baith Mohamed, Gerald Quirchmayr, and Erich Schikuta. Ontology-Based Security Requirements Framework for Current and Future Vehicles. Vol. Data Science and Big Data Analytics in Smart Environments. CRC Press Taylor & Francis Group. - (Accepted)*
- *Schikuta, Erich, Abdelkader Magdy, and A. Baith Mohamed. 2016. "A Framework for Ontology Based Management of Neural Network as a Service." Pp. 236–243 in Neural Information Processing, edited by A. Hirose, S. Ozawa, K. Doya, K. Ikeda, M. Lee, and D. Liu. Cham: Springer International Publishing.*
- *Shaaban, Abdelkader Magdy, Christoph Schmittner, Gerald Quirchmayr, A. Baith Mohamed, Thomas Gruber, and Erich Schikuta. 2019. "Toward the Ontology-Based Security Verification and Validation Model for the Vehicular Domain." Pp. 521–529 in Neural Information Processing, edited by T. Gedeon, K. W. Wong, and M. Lee. Cham: Springer International Publishing.*
- *Schikuta, Erich, Abdelkader Magdy, Irfan Ul Haq, A. Baith Mohamed, Benedikt Pittl, and Werner Mach. 2017. "Searching the Sky for Neural Networks." Pp. 167–178 in Advances in Computational Intelligence, edited by I. Rojas, G. Joya, and A. Catala. Cham: Springer International Publishing.*
- *Magdy, Abdelkader, A. Baith Mohamed, Gerald Quirchmayr, and Erich Schikuta. 2017. "Towards a Security and Privacy Protection Model for Semantic Query Engines." Pp.*

198–207 in Proceedings of the 19th International Conference on Information Integration and Web-Based Applications &; Services, iiWAS '17. New York, NY, USA: Association for Computing Machinery.

2.1 Safety in Modern Vehicles

The vehicular industry is rapidly evolving from mechanical units working on gears and shafts to electronic components interacting using different communication protocols. Semi and fully autonomous vehicles are an expressive example of integrating advanced technology within the vehicles manufacturing. The integration of modern technologies with the vehicle provides many exciting features such as Tesla [1] Autopilot, which can control the steering, the speed, and the brakes of the vehicle automatically within the vehicle's lane [Tes20]. Autonomous vehicle technology is one of the most vivid examples of new IoT applications on the internet. Netscribes market research expects the international car IoT market to cross $106,32 billion by 2023 [Net18].

According to the National Highway Traffic Safety Administration (NHTSA) [2], " *The safety benefits of automated vehicles are paramount*." Automated vehicles will help to reduce the injury rate and save lives, as 94% of serious accidents are human mistakes [NHT17]. The Society of Automotive Engineers (SAE) [3], defines six levels of automation in the range between (0) to (5), where the level (0) is non-driving automation, and (5) is the full driving automation [SAE18]. The important distinction among these levels exists between level 2, where the human performs part of the driving function, and level 3, which carries out the whole dynamic driving process using the automated driving system [Int14]. The technological description explains the functions of the machine, and what actions the driver needs to perform [dAe15]. Figure 2.1 depicts the six levels of automation, driving from level (0) to level (5). As discussed in [Syn19], the six levels of automation are:

- **Level 0 (No Driving Automation):** driving is performed manually. The driver offers "driving mission," while systems assist the driver.
- **Level 1 (Driver Assistance):** this is considered the lowest level of automation. The vehicle has only one automatic driver assistance system, such as accelerating or steering.
- **Level 2 (Partial Driving Automation):** the level of automation includes Advanced Driver Support Services (ADAS) that is considered a short self-driving because a driver always takes over the control of the vehicle from time to time. For instance, Tesla Autopilot and Cadillac Super Cruise are the best examples of level 2 automation.
- **Level 3 (Conditional Driving Automation):** vehicles of level 3 are equipped with "environmental detection" and can make decisions on their own, but they still necessitate a watchful driver.
- **Level 4 (High Driving Automation):** vehicles of level 4 can intervene when something goes wrong, or system failure occurs. These vehicles work in self-driving mode but in a defined environment.

[1] https://www.tesla.com/en_eu
[2] https://www.nhtsa.gov
[3] https://www.sae.org

- **Level 5 (Full Driving Automation):** this level requires no driver attention. Vehicles of level 5 can interact as an experienced driver; they can move to anywhere without geofencing.

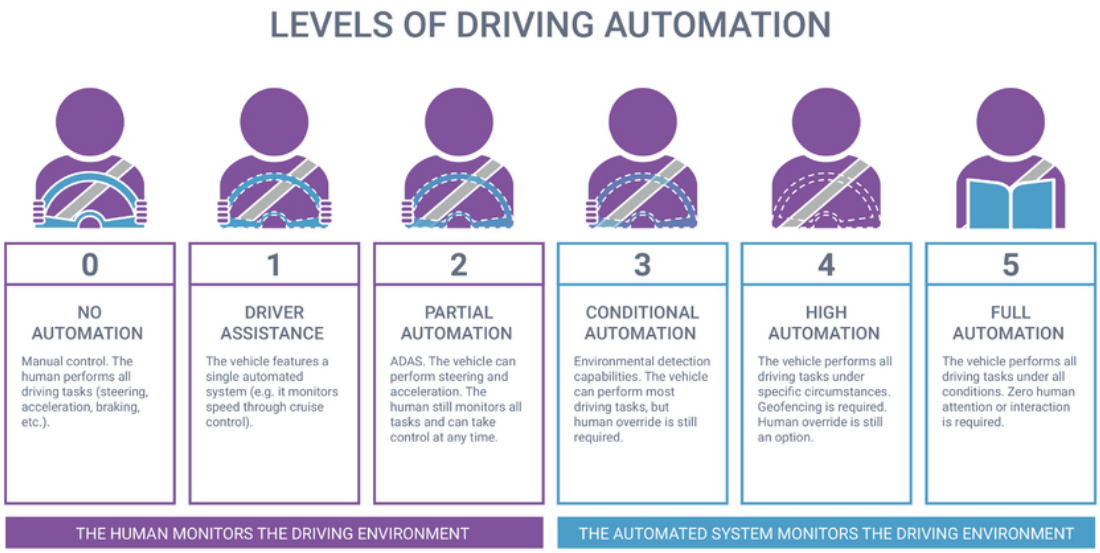


Figure 2.1: Six levels of automation driving [Syn19]

According to the World Bank, in developing economies, traffic jams can cost up to 5% of annual Gross Domestic Product (GDP). In developed economies, road congestion can cost 0.3-0.5% of their annual GDP [Car17]. Ten ways can enhance the autonomous driving of vehicles, as discussed in [BW15]. Figure 2.2 illustrates the timeline of the autonomous driving starting from 2015 to 2050.

The figure classifies autonomous vehicles into three stages; each represents a modern age of autonomous development evolution.

Era one: autonomous vehicles are being developed: In non-road applications, autonomous vehicles are already selected for environmental control, such as mining and farming. The benefits of incorporating autonomous vehicles in these fields include saving labour costs and reducing carbon dioxide (CO2). In parallel with the autonomous vehicles, there are different transport mobilities integrated with the road, such as car-sharing and peer-to-peer rental cars, according to the autonomous driving growth. Also, the automaker's vision will undoubtedly be defined and communicated in the coming years of autonomous vehicles [BW15].

Era two: early-adoption step: The autonomous vehicle development will change the car service landscape. For example, 80% of car services in Germany were entirely independent of the Original Equipment Manufacturer (OEM). From the safety-critical, it may be advantageous to limit the use of the original equipment of the OEMs to maintain and repair autonomous vehicles. Autonomous vehicles may also change insurance companies' strategies. These companies provide

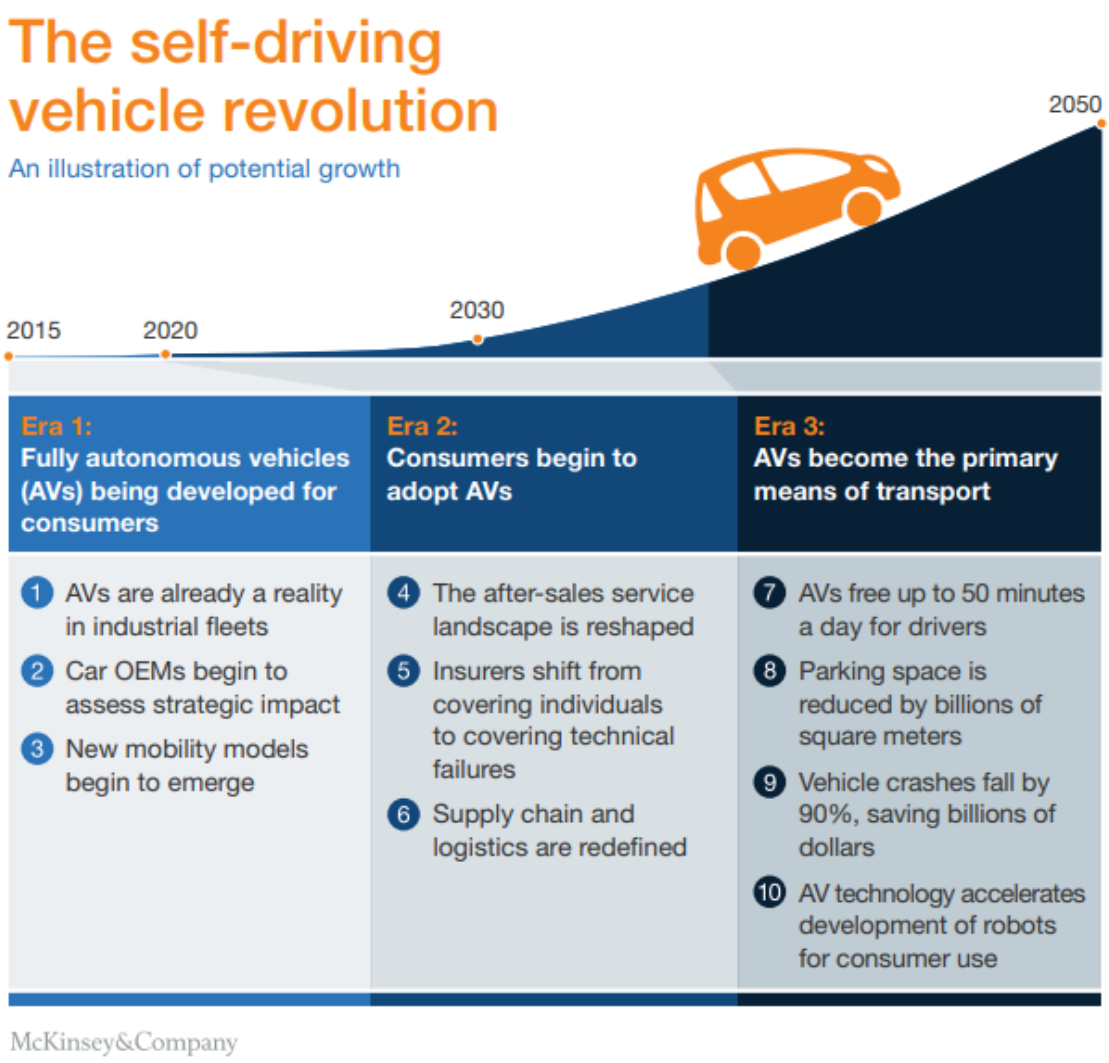


Figure 2.2: Revolution of the autonomous vehicle [BW15]

consumer insurance in accident cases caused by human error. However, this strategy will change to be entirely concentrated on the manufacturer, and the probability of technological failure of the autonmous vehicles. This would reduce emphasis on consumers and increase emphasis on OEMs such as the insurance for shipping companies. Moreover, one of the autonomous vehicle's specific features is to help optimize the industry's potential supply chains and the logistics operations that can minimize labour costs, volumes, and stocks using smart distribution methodology [BW15].

Era three: autonomous vehicles become the primary form of transport: Autonomous vehicles may save driver's time, allowing them to spend every day to use in other activities, such as work, relaxation, or entertainment. Parking also becomes more convenient by using autonomous vehicles, as this reduces the need for the space required for parking. For example, passengers do not need to save space for door oppening, which may be a factor in saving parking area infrastructure. Accident rates will be dropped using autonomous vehicles. For example, in 2012, the estimated annual cost of road accidents to the U.S. economy was $212 billion. However, using

autonomous vehicles and Advanced Driver Assistance Systems (ADAS), incidents rates can be minimized by up to 90%, which saves about $190 billion a year. Moreover, autonomous vehicles can accelerate robotics production for market applications [BW15].

Safety engineering is used to create safe systems with a focus on reliable parts [SMS14]. Cornell Aeronautical Laboratory, New York, was the first academic research institute to enhance vehicular safety. The critical demonstration of their study is the importance of seat belts and padded dashboards [SRI16]. The vehicular domain contains a collection of developed safety standards that are used to assure that all safety risks are mitigated to a tolerable level. In addition, these standards are applied to design systems and safety-critical components [SM14]. Faults or deficiencies in either reliability for safety or computer-based automotive systems can cause severe consequences. At least one death is attributed to defects in computer systems in vehicles. Roughly 500 injuries and fatalities were allegedly attributed to some faulty vehicle designs by the same OEM [Koo18].

The ISO 26262 [ISO18] is a worldwide accepted standard for automotive electrical and/or electronic systems design and development [emb19]. It is the first complete standard to address the functional safety of the Electrical/Electronic (E/E) components in the current automotive domain [Adm16]. The standard is applicable for vehicles with weight less than 3500 Kg, presenting guidance and recommendations through the product development life-cycle starting from the conceptual level until the decommissioning [Alh17]. The standard is a framework that combines functional safety into the development life cycle of the automotive industry [emb19]. According to the ISO 26262 safety standard, the life-cycle of an automotive component begins with determining where the system is being used and how important it is for automotive safety [Sys19]. According to ISO 26262, a Hazard Analysis and Risk Assessment (HARA) is used to measure the critically of the system. To ensure functional safety in vehicular units, HARA findings are used in other sub-development phases [SBRM17]. The standard has several parts, discussed in [Alh17], as follows:

- **Part 1:** Vocabulary.
- **Part 2:** Management of functional safety.
- **Part 3:** Concept phase.
- **Part 4:** Product development at the system level.
- **Part 5:** Product development at the hardware level.
- **Part 6:** Product development at the software level.
- **Part 7:** Production and operation.
- **Part 8:** Supporting processes.
- **Part 9:** Automotive Safety Integrity Level (ASIL).
- **Part 10:** Guidelines on ISO 26262.

Figure 2.3 illustrates the overall structure of the ISO 26262 standard. A V-model is defined as a reference process model in the standard for product development [ISO20a].

The standard sets out four potential safety impact levels, known as Automotive Safety Integrity Level (ASIL). ASIL is one of four levels (e.g., A, B, C, and D) to identify safety measures and

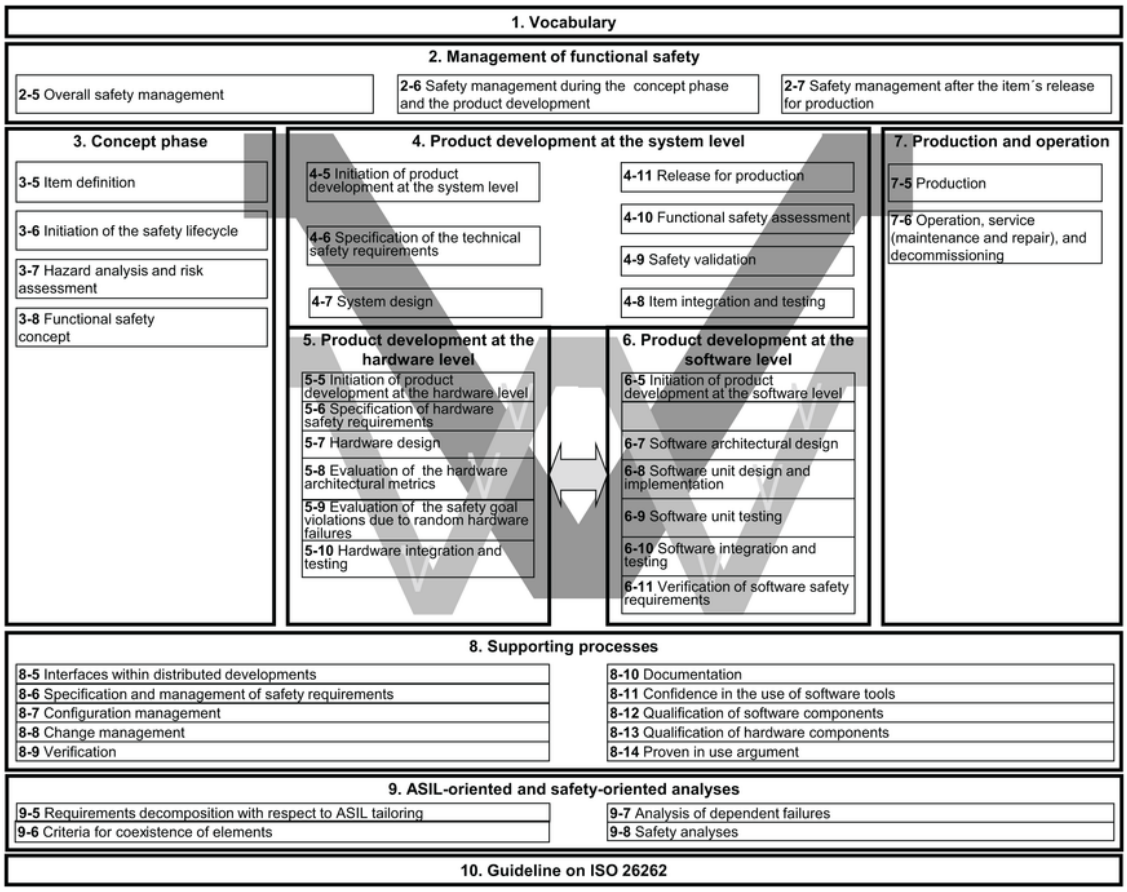


Figure 2.3: ISO 26262 Overview [ISO20a]

requirements needed by the system to prevent critical residual risk. Level D is defined as the highest level, where A is the lowest [JHL15]. After the object is identified, the hazard analysis and risk assessment is performed, referring to ASIL. The ASIL calculation is based on the Equation 2.1, as represented in [Alh17]:

$$Risk = Severity * Frequency * Controllability \tag{2.1}$$

The equation evaluates the hazardous event according to the following parameters [Alh17]:

- **Frequency:** probability of a situation to happen.
- **Impact:** the severity of outcomes if a hazard should occur.
- **Controllability:** the degree of driver controllability of the vehicle.

After the SAIL is measured, the system's safety target is specified, which is considered the top-level safety requirement based on the HARA findings [Alh17]. The safety goals are more critical than the functionality of the automotive unit. For example, safety goals related to the battery are more critical than the battery itself. An overheating of battery when the vehicle's speed

is less than 10 Km/h is not considered a serious situation (i.e., ASIL-A). The ASIL-C could be estimated when the speed of the vehicle is more than 50 km/h. Therefore, ASIL estimation is considered a critical task in the development process of automotive applications. Increasing the use of ECU, sensors, and actuators in the automotive domain require ensuring functional product safety at any point in the product lifecycle [emb18].

The American Automobile Association (AAA) [4] performed tests on cars with pedestrian detection warnings and automatic emergency braking using dummy pedestrians. The automobiles hit the dummies in 60% of the trials. When the researchers replaced the adult pedestrian dummies with a child-sized dummies, the rate at which the automobiles hit the dummies increased to 89%. [AJH19]. Modern pedestrian detection systems can inform the driver when there is a serious risk of a collision through an audible, visual, or haptic signal, where the driver is considered the actuator that is controlled by haptic warning [Gus09]. Roughly 56% of 2018 model year cars are fitted with pedestrian detection capability and automatic emergency braking as stated in the research report by AAA [AAA19]. In 2016, the National Highway Traffic Safety Administration (NHTSA) [5] published technical support to compare and assess the applicable safety standards (e.g. ISO 26262 [ISO18], MIL-STD-882E [AS12], DO-178C [WDRTCfA11], FMVSS [ADM99], AUTOSAR [AUT03], and MISRA C [MIS04]) [Adm16].

Integrating new technologies with the automotive industry opens the way to cybersecurity as one of the essential issues that need to be incorporated into the lifecycle of automotive development. Lack of security protection mechanisms in vehicular design may lead to different ways of executing malicious attacks against the vehicular network. These attacks could have various negative consequences, such as compromising the safe operation of the vehicle. Protecting the vehicle from various cyber-attack incidents is a crucial process, which could help in decreasing vehicle safety hazards. The next section addresses critical aspects of cybersecurity protection in the automotive domain, including a real hack scenario in a vehicle. This example provides further evidence that automotive cybersecurity is not only about protecting private data, but also how it can now help to minimize accident rates.

2.2 Cybersecurity Challenges in the Automotive Domain

Modern automotive architectures and software are considered the best example of cyber-physical systems (CPS) [GSM+12]. The CPS are physical objects with a virtual representation, which can connect through the worldwide network with other CPS [PMRMP18]. The vehicles are trending towards more difficulty and connectivity; combining these trends would require much more emphasis on cybersecurity. Modern cars do not come with keyboards but contain numerous communication protocols, electrical/electronic units, and operating systems. Therefore, these cars are considered a collection of computers with wheels attached, which could be a daunting hacking target [Smi16]. For example, autonomous vehicles are considered computers in motion; any such malicious attack may lead to critical exposure, making security much more essential issue. Confidentiality, Integrity, and Availability (CIA) must be integrated with all development phases in the automotive industry [AVL20].

All road transport actors, road infrastructure, and authorities should cooperate in developing fully or highly self-managed vehicles. These parts are influenced by an integrated infrastructure system, which requires new reliable communication approaches that enable vehicle-to-vehicle

[4] https://www.aaa.com/International/
[5] https://www.nhtsa.gov/

(V2V), and vehicle-to-infrastructure (V2X) communication. Reliable communication is considered the critical requirement for processing and speeding up the development of different motor vehicle systems [SLAMH18]. The V2X is considered the advanced technology towards a new traffic safety edge, and the gateway to interact with vehicles and road infrastructure together. Accordingly, new security standards must be considered to prevent electronic vehicle units from being compromised by various malicious attacks.

The project EVITA (E-safety vehicle intrusion protected applications) [6] was launched in 2008 to design, verify, and prototype security building blocks for on-board automotive networks. The project developed a unique onboard automotive risk analysis approach. EVITA defined on-board vehicle security requirements, and according to these security requirements, EVITA developed secure architecture and secure communication protocols for the automotive on-board network [EVI08]. The new vehicle generation can run several applications simultaneously, even if they are not built from a corresponding OEM. Therefore, to run applications safely and securely, these applications must be run on virtual machines to ensure complete isolation from possible failure or malicious actions. The critical point is the running applications typically use the same vehicle resources as the display, interfaces, and vehicle information. Therefore, it is highly demanded to ensure that the running applications downloaded from the internet do not damage vehicle assets (e.g., information) or impact vehicle safety adversely. Moreover, developing a secure platform is essential to run different applications from different vehicle manufacturers sharing the same vehicle resources. The Open VEhiculaR SEcurE platform (OVERSEE) [7] project is launched was 2010 to provide internal and external vehicle interfaces for automotive applications. OVERSEE developed an open, standardized software and communication framework to open up a vast opportunity to develop vehicle applications. OVERSEE included that applications cannot adversely affect each other or affect any internal vehicle system units. The security protection frameworks followed in the OVERSEE platform aimed to manage the inferences from applications, human error, or malicious acts [OVE10]. The HEAling Vulnerabilities to ENhance Software Security and Safety (HEAVENS) project was launched in 2013 to identify security weaknesses in the automotive system and develop new methodologies to evaluate security. The project's primary goal is to define a set of security measures for the asset owner to provide a protection mechanism for the assets. This minimizes the risk relevant to these security weaknesses, which could be exploited by threats. HEAVENS introduced fundamental dependability and security concepts, including security attributes and security objectives [Ols16]. The project used eight security attributes (i.e., confidentiality, integrity, availability, authenticity, authorization, non-repudiation, privacy, and freshness) as an extension of the CIA triad (i.e., Confidentiality, Integrity, Availability). In addition, HEAVENS integrated a set of security objectives, aiming to address the identified threats to meet the security policies and assumptions [LI16].

In 2016, the SAE released "Cybersecurity Guidebook for Cyber-Physical Vehicle Systems - J3061." This guidebook gives a set of high-level standards to govern cybersecurity in cyber-physical vehicle systems. This guide covers the following key points [SAE16]:

- Defines a complete process lifecycle, which could be adapted by organizations into development processes to integrate cybersecurity into cyber-physical vehicle systems.
- Addresses the tools and methodologies used commonly to verify and validate cyber-physical vehicle systems.

[6] https://www.evita-project.org/index.html
[7] https://www.oversee-project.com

- Gives a base-line principle on the cybersecurity for the vehicle system.

SAE J3061 is focused on other existing safety engineering and secure system methodologies that have a relationship with functional safety standard ISO 26262; therefore, J3061 is strongly influenced by ISO 26262 [SMR+16]. The guidance divides the system's lifecycle into different phases, including concept, product development, production, operation, and service [SMR+16]. The J3061 aims to adjust the processes between safety and security to organize the two engineering processes [MS16] [SMR+16]. Instead of HARA in ISO 26262, the J3061 describes Threat Analysis and Risk Assessment (TARA) activities to identify the possible threats and estimate the related risk. As described in Section 2.1, ISO 26262 implemented HARA to estimate the vehicle system's safety hazard level and cybersecurity; TARA is applied to identify potential cyber threats and intends to estimate associated risks. The best technique for detecting threats and defining the relevant risks is the threat modelling approach, as discussed by in [MS16]. The next section discusses different available methods for the threat modelling technique. One of these techniques is integrated within this thesis to identify potential threats that negatively affect vehicular networks.

2.2.1 Threat Modeling Methods

Threat modelling technique is an approach used for detecting the security threats and vulnerabilities in the system model; it defines the most cost-effective security measures to address threats [AGK15]. Threat modelling is the building block in the system engineering process to identify possible threats and give a clear vision toward which security mitigation approaches need to be integrated within the system to reduce the overall risk. It aims to deal with this issue by defining an abstract model of threats applied to a system to recognize potential threats to the system [ESSK19]. Ref [MS16] shows a conceptual representation of implementing a threat modelling technique into the automotive security analysis process. Figure 2.4 illustrates this representation of threat modelling in the automotive domain.

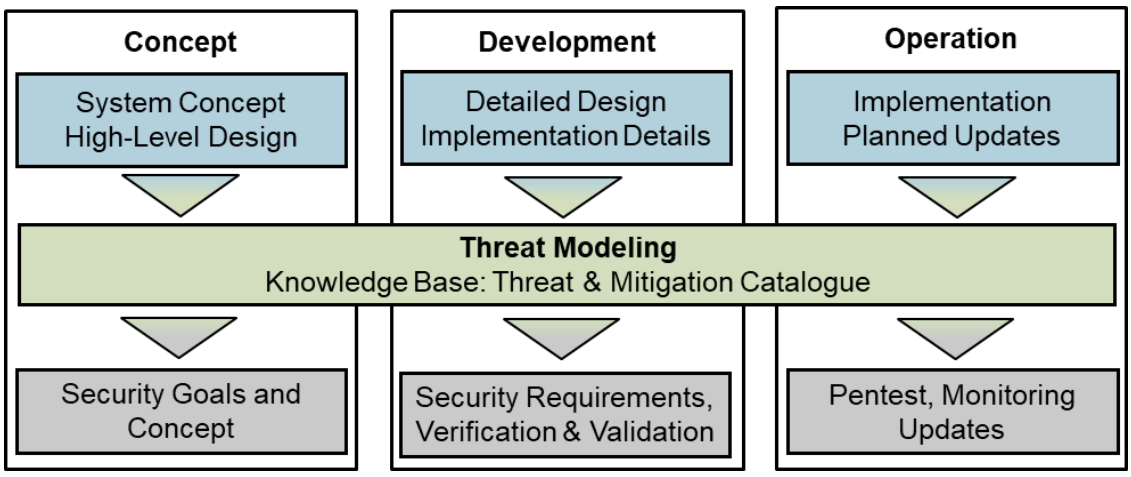


Figure 2.4: An overview on the threat modeling technique in the automotive domain, updated design based on [MS16]

The above figure shows that the threat modelling technique can be integrated into any automotive lifecycle phase. As mentioned [MS16], threat modelling can be incorporated into the concept process to present high-level security requirements. It can also be integrated into the

development phase using the system design specification with all relevant details to determine security requirements and identify vulnerabilities in system architecture. Then it can be used to verify and validate security requirements. In the third phase (i.e., Operation), threat modelling acts as a supporting technique on the finished automotive components and systems, which provides a checklist of matters to prioritize in the pen-testing process. The threat modelling technique is an iterative process according to continuous change in the development and modification of system design [MS16]. In general, threat modelling has a set of defined sequence actions that are described in [STH+19], are described as follows:

1. Define the system model structure with all related information (i.e., system specification).
2. Model the potential adversaries with relevant techniques and actions.
3. Apply the threat analysis to the system model to define the potential exiting threats.
4. Estimate the risk of each threat to select the applicable risk treatment method.
5. Update the model of the system with security countermeasures.
6. Repeat step 3 to check the effectiveness of the applied security measures and check if there are any missing potential threats that are not yet identified.

Ref [She18] summarizes different threat modelling methods; with no particular method proposed to be superior to any other. This section introduces the most common threat modelling methods applicable in the automotive domain.

STRIDE

STRIDE is the abbreviation of the **S**poofing, **T**ampering, **R**epudiation, **I**nformation Disclosure, **D**enial of Service, and **E**levation of Privilege. It was invented in 1999 and adopted by Microsoft [8] in 2002 [She18]. The threats categories are discussed in [AGK15] as follows:

- **S**poofing: Trying to get unauthorized access through false identity.
- **T**ampering: Intending to modify data by an unauthorized method.
- **R**epudiation: Denying an action that is performed by legal/illegal user.
- **I**nformation Disclosure: When private data has been revealed by an undesirable method.
- **D**enial of Service: When an action is taken to make a specific service, system, or application unavailable.
- **E**levation of Privilege: If a user with restricted authorization claims has a higher privilege than they hold.

STRIDE is based on the Data-flow Diagram (DFD) to identify the system entities, events, and the different levels of system zones [She18]. As discussed in [STH+19], there are five basic annotations are used to describe the system elements using DFD:

[8] www.microsoft.com

- **Process:** represents the system elements, input, output, and actions.
- **Data Store:** represents a data storage or databases, which indicate the sources of the system data.
- **Data flow:** represents the flow of the information between the different system entities that could represent a communication protocol.
- **External Interactions:** indicates external elements that interact externally to the system elements.
- **Trust Boundaries:** are used to divide the system structure into separate trust zones.

The STRIDE method identifies potential threats in the design phase of any software and hardware application. STRIDE has two variant approaches: per-element and per-interaction for performing threat modeling. [KKA17].

STRIDE per-Element: this approach observes the threats according to relevant elements in the diagram. It focuses on a particular set of threats for each specified system element. According to Table 2.1, Microsoft uses this table as a core of its security development lifecycle [Sho14] [SS04].

Table 2.1: The Microsoft's STRIDE-Per-Element [Sho14]

	S	T	R	I	D	E
External Entity	X		X			
Process	X	X	X	X	X	X
Data Flow		X		X	X	
Data Store		X	?	X	X	

The "X" in the table represents the class of threat that could be triggered by the element category. "?" means that the logging data store element is applied in addressing the repudiation, but in some cases, the logs could be used as a way for repudiation attack. For example, suppose there is a network data stream, and an attacker connects to the same network. In this case, attackers can read, modify, or send a flood of packets to deny a particular service [Sho14].

STRIDE per-interaction: is the second approach of the STRIDE method that identifies threats against the considering tuples: origin (i.e., source), destination (i.e., target), and interaction. This approach gets the same number of threats as STRIDE-per-element. However, the STRIDE per-interaction is considered more manageable because the threats could be easier to understand than the STRIDE-per-element. This approach needs a reference chart of software to obtain the results. [Sho14][SS04].

PASTA

Ref [She18] discusses another method of threat modeling called PASTA. The Process for Attack Simulation and Threat Analysis (PASTA) is a threat modeling method used to teach people interested in cybersecurity how to analyze threats and attacks. The PASTA approach outlines essential steps for analyzing threats and attacks. It also builds countermeasures to reduce business impact [noa12]. In the PASTA methodology the impact could be determined in the early stages

of the analysis process rather than in the evaluation phase. Calculating the PASTA impact earlier gives more knowledge about the consequences of impacts than in the threat analysis process [OWA20]. The PASTA method has seven phases for analyzing threats and simulating attacks on an application [LI16]. Each of these phases has multiple activities, as illustrated in figure 2.5 [She18].

Figure 2.5: The seven phases of PASTA method [She18]

Each stage in the above figure contains particular objectives; objectives of each stage are presented in [noa12], as follows:

- **Define Objectives**: this stage focuses on understanding the impact, defining the risk mitigations, and identifying the security requirement to achieve these objectives.
- **Define Technical Scope**: the second stage aims to document all application profiles and to gather relevant design blueprints of all use-cases and transactions such as sequence documents, transaction flow, and architecture design document.

- **Application Decomposition**: this stage aims to assist threat analysis in mitigating threats. Therefore, it intends to identify application security controls with the application's function and transaction dependencies.

- **Threat Analysis**: this stage collects threats and all relevant attack information from threat intelligence and other sources.

- **Vulnerabilities & Weaknesses Analysis**: at this point, the threat analyst investigates possible vulnerabilities and exiting security flaws discovered either by pen-testing (i.e., back box) or by analyzing source code (i.e., white box). A correlation framework is then applied to map vulnerabilities to threats to identify which vulnerabilities could be exploited by a particular threat.

- **Attack Modeling**: in this stage, the design flaws and exiting security control gaps are identified at the application architecture level and the functional level. This can be done by analyzing and simulating the attack scenarios similar to what attackers do, including analyzing the attack using attack trees and analyzing the violation of security controls using the "use and abuse cases" technique.

- **Risk & Impact Analysis**: the last stage in the PASTA methodology is responsible for analyzing the impacts and risks, and formulates the risk reduction plan for mitigating risks. The categorization and estimation of risk factors such as threats, attacks, vulnerabilities, and business impact are essential for the risk analysis process. The risk mitigation strategy includes preventive and detective controls and new governance methods such as risk-based testing, fraud detection, threat analysis, and cyber-intelligence.

CVSS

The CVSS (Common Vulnerability Scoring Systems) is another threat modelling method, discussed in [She18]. CVSS is produced by the National Institute of Standards and Technology (NIST) [9] and maintained by Forum of Incident Response and Security Teams (FIRST) [10] [She18]. It provides an open framework to estimate the level of severity of the IT vulnerabilities. Each IT vulnerability has a severity score (i.e., CVSS) that ranges from 0.0 to 10.0 [GB11]. This estimation depends on three metrics (i.e., Basic, Temporal, and Environmental), Figure 2.6 depicts the metrics of the CVSS.

The three metric groups are discussed in [FIR20] as follows:

- **Basic Group**: metrics represent characteristics of a vulnerable thing that are constant over time and cross the user environment. This group has two sets of metrics:
  - **Exploitability Metrics**: define the technical means by which vulnerability can be exploited. These metrics represent the characteristics of a vulnerable node (i.e., software, module, driver, etc.).
  - **Impact Metrics**: represent the direct consequence of real successful exploitation and represent the outcome of the impact.

[9] https://www.nist.gov
[10] https://www.first.org/

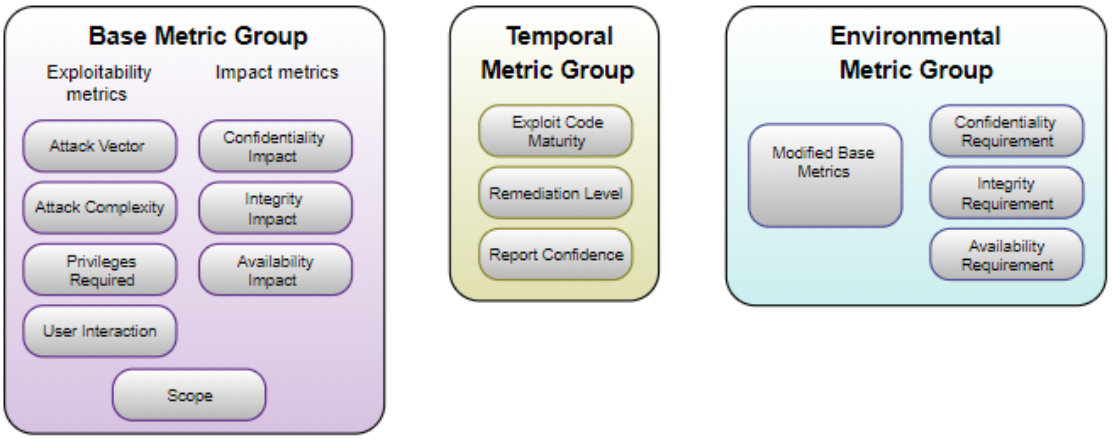


Figure 2.6: The three metrics of the CVSS [FIR20]

- **Temporal Group**: this group represents the vulnerability characteristics that may change over time but not over the user environments.
- **Environmental Group**: this group describes the properties of a vulnerability related to the environment of users.

The analyst assigns parameter values for each metric in order to calculate the CVSS score [She18]. All almost known vulnerabilities scores are estimated by the National Vulnerability Database (NVD) [11] [NVD20b]. The NVD provides an online CVSS score calculator to allow the user to add temporal and environmental parameters for the vulnerability scoring. The calculator can be found in [NVD20a].

Attack Trees

The attack trees method is a formal approach for analyzing the security of a system and its relevant subsystems [Bru99]. This approach is considered one of the oldest and most common security analysis techniques applied to cyber systems, physical systems, and cyber-physical systems [She18]. The attack trees approach has three different types of nodes; as discussed in [SRAS19], these nodes are defined as follows:

- **First type**: the deepest level of nodes (i.e., leaf nodes) play an essential role in the initiation of the attack path, which consider the actions or attacks facts that can be started without any other requirements.
- **Second type**: the OR nodes; when OR nodes describe an attack, this means any child nodes in the attack tree represent a possible way/path of an attack to happen. Simply as the OR operator in the propositional logic, the whole formula is True if any or all of the formula clauses are True.
- **Third type**: represented by AND nodes. When nodes represent an attack, it means all child nodes must occur to express that the attack happens. Similar to the AND operator in the logical operations, the whole formula is True if all the operands are True as well.

[11] https://nvd.nist.gov

A simple attack tree example is discussed in [Bru99], against a physical safe. Figure 2.7 illustrates the attack tree of this scenario; the goal of the attack is the physical safe. Hence, it is represented as the tree's root, where the attack approaches are represented as the leaf nodes.

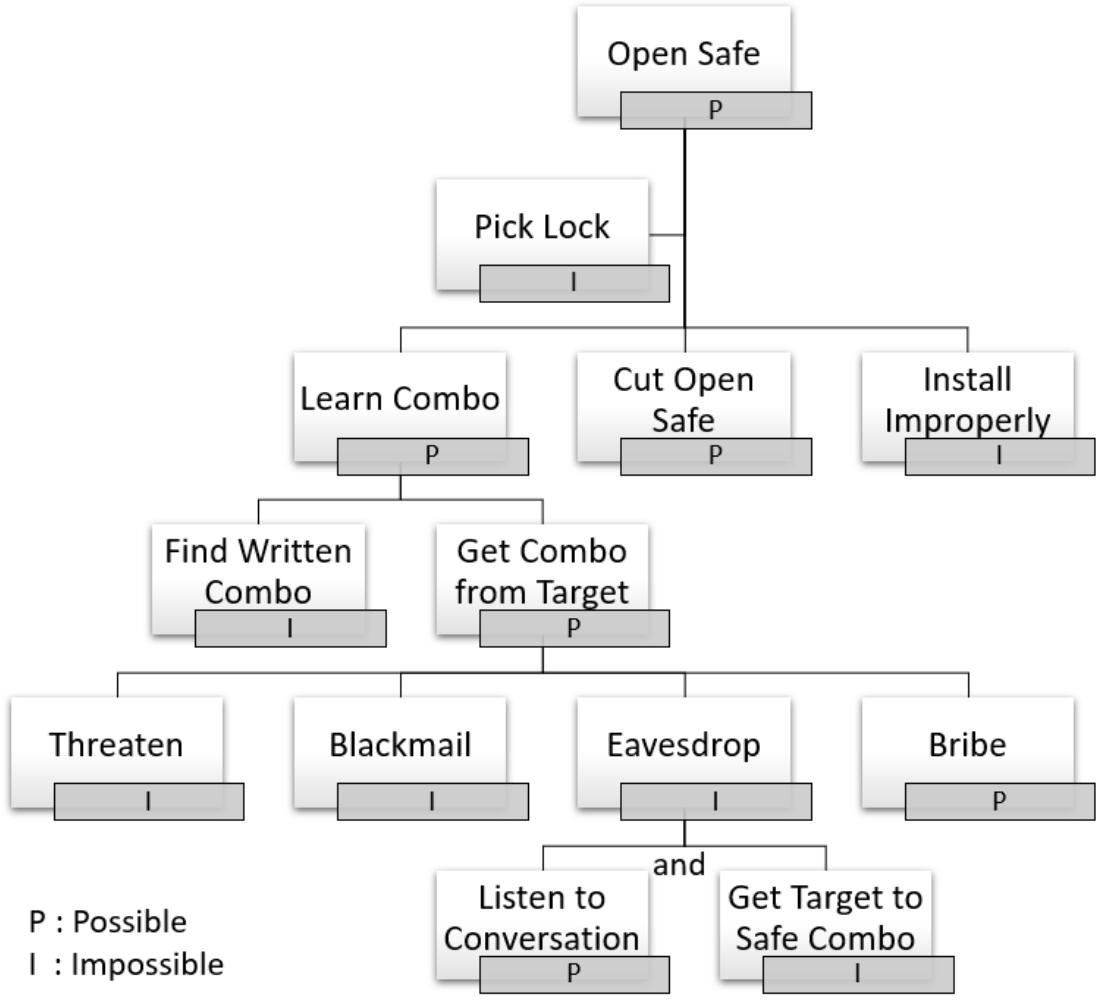


Figure 2.7: The tree nodes structure of physical safe attack, according to the example in [Bru99]

In order to open the safe, the attack has different alternative options (e.g., pick the lock, learn the combination of the key, cut open the safe, or improperly install the safe). Learn the combination option has two methods: either getting the combination from the owner of the safe or getting that in a written form. The threaten blackmail, eavesdropping, or bribe are the four methods to get the combination from the owner. The nodes of the trees could be OR and AND nodes. As discussed earlier, the OR node is possible if any of the child nodes are possible to happen. However, the AND nodes are used here to represent multiple nodes that should be possible to achieve the same goal [Bru99].

OCTAVE

The Operationally Critical Threat, Asset, and Vulnerability Evaluation (OCTAVE) is a risk-based strategic method for evaluating and planning cybersecurity methods. OCTAVE was created by the CERT Division [12] in 2003 and then refined in 2005 [She18].

OCTAVE is a flexible evaluation approach that can be handled for most organizations, which are focusing on strategic and practice issues. In using the OCTAVE approach, a team of people from the operational units and the information technology can work mutually to address the organization's security demands. A three-phased approach defines the risk evaluation of information security. OCTAVE organizes these phases to enable personnel to assemble a comprehensive view of an organization's information security demands [ADSW03]. Figure 2.8 illustrates the OCTAVE phases.

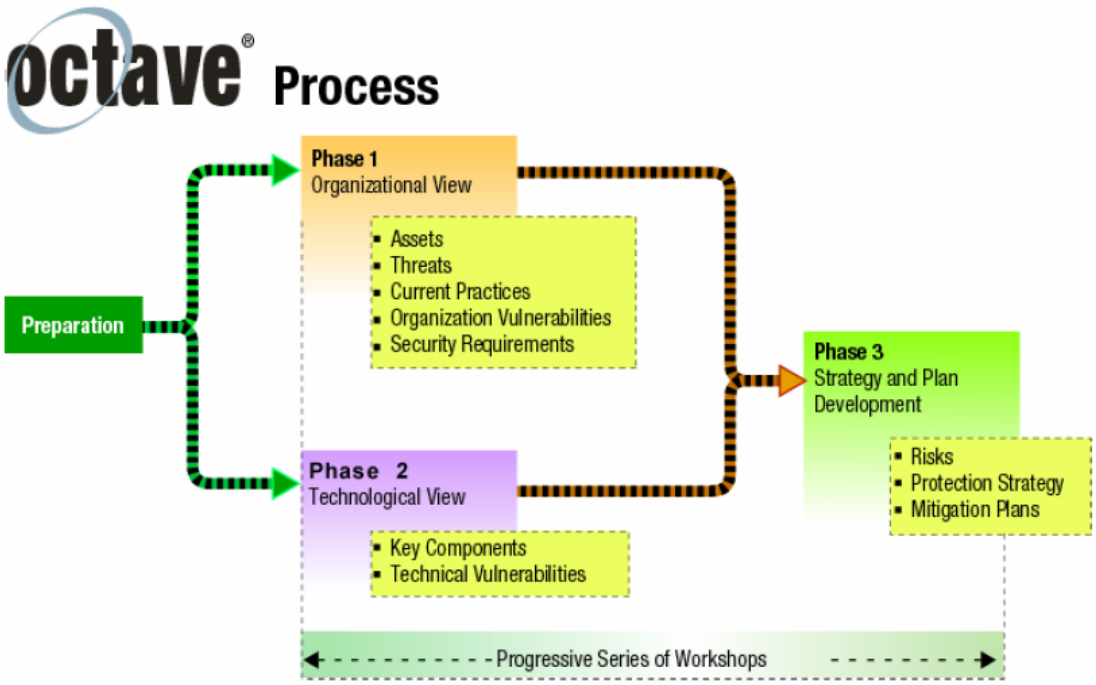


Figure 2.8: The OCTAVE phases [ADSW03]

- **Phase one: Build Asset-Based Threat Profiles**: The analysis team determines the organization's assets to define the most important assets to the organization (i.e., critical assets). Then the team describes the relevant security requirements for each of these critical assets. At the end of this analysis phase, the team defines related threats for each asset by creating a threat profile for the asset.

- **Phase two: Identify Infrastructure Vulnerabilities**: The analysis team examines the paths of the network access, identifying classes of information technology components relevant to the previously defined asset in phase one.

[12] https://www.sei.cmu.edu/about/divisions/cert/index.cfm

- **Phase three: Develop Security Strategy and Plans**: In this phase, the analysis team evaluates the risks to the organization's critical assets. The analysis team also creates a protection strategy and mitigation plan for the organization to address the existing risks to critical assets.

2.2.2 Threat Modeling Approach in the Automotive Domain

Threat modeling deals with the automotive domain by building an abstract model of threats, which are applied to vehicular design to recognize the potential threats to the vehicular system. Ref [AMC20] introduces ThreatGet [13] a novel threat modeling methodology was specifically developed for the automotive sector. The tool automates the threat analysis process and supports the security mitigation for reducing the overall risk in the vehicular domain. ThreatGet is a part of this thesis; we aimed to introduce a new effective threat modeling methodology to be integrated into the early stages of the vehicle engineering process. It is developed to identify potential threats in the vehicle design and quickly evaluate the relevant risk to estimate the overall risk in the vehicle design.

ThreatGet is introduced to help in identifying potential threats in the system model in the early stages of the development process. It is an Enterprise Architect (EA) based tool is developed by the AIT Austrian Institute of Technology [14]. EA by Sparx Systems [15] is a commonly used platform for model-based systems engineering. The tool aims to analyze the entities, artifacts, and links in the model design to identify the potential threats and risk assessment [ESSK19]. It contains three main parts; the first part is the EA Add-in, the second and the third parts are web-based front-end and back-end sides. Figure 2.9 illustrates the main building blocks of ThreatGet.

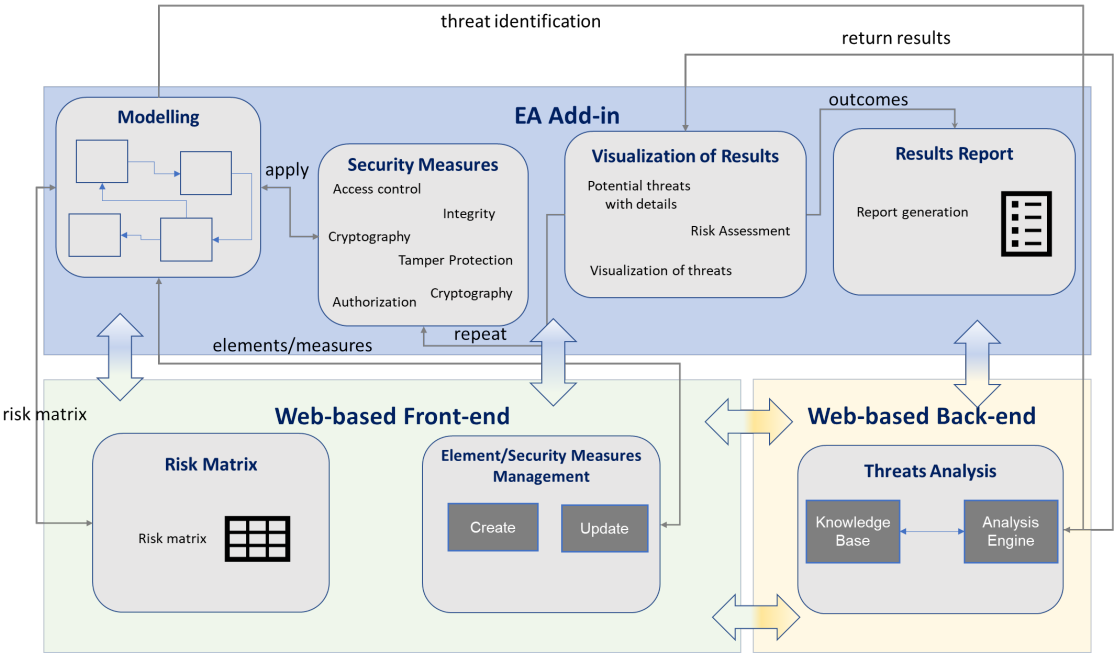


Figure 2.9: The building blocks of ThreatGet

[13] https://www.threatget.com
[14] https://www.ait.ac.at/en/
[15] https://sparxsystems.com

EA Add-in The ThreatGet Add-in can be easily integrated into the EA on the local user machine. The Add-in supports four separate phases in the threat modeling process, starting from modeling the system design until reporting the result in full documentation. The user collects all information related to the system structure and creates a diagram of the system model similar to system design, but focused on security. This model illustrates a realistic image of the interaction between the entities in the vehicular domain. Furthermore, this model is considered a living document containing the tracked changes and is kept up-to-date [Smi16]. ThreatGet supports the most common vehicular units and communication protocols that can be used to design a wide range of scenarios to define the internal interactions between the vehicular components. Plus, it supports different components that can be used to define the external interaction between the vehicle and the external environment, such as roadside units (RSUs).

In addition, the add-in plays an integral role in the threat modeling process for managing security measures, which are used to examine how vehicle units and their interactions are secure against a wide range of potential threats. Furthermore, the user selects the applicable security properties that should be integrated into the vehicular units as a protective methodology for keeping the vehicle secure. These measures are used in ThreatGet to describe the security and analysis of related properties for elements [ESSK19]. ThreatGet provides an exhaustive list of security measures to select the most suitable for the vehicle. The user selects a set of these measures to be a reflective image of the vehicle design; then, these measures are used in the threat analysis process to check how secure the vehicle model is and identify if there are remaining security weaknesses require attention.

ThreatGet covers a wide range of threats based on the STRIDE model. ThreatGet tracks all vehicle model connections to identify possible threats within the internal vehicular structure (i.e., source, target, and communication flow). Also, ThreatGet performs a risk assessment process, which gives an overview of the overall risk. A risk assessment process is an essential approach for estimating and evaluating the risk severity level for each threat [RPMS17]. This process is based on impact and likelihood parameter values [SSG+19]. ThreatGet supports different degrees of likelihood and impact. Users can handle these levels to determine the most significant risk assessment approach in different applications such as automotive, railways, Internet-of-Things (IoT), CPS, etc. ThreatGet defines five levels of risk (e.g., from 1 to 5); level 5 is the highest severity level (e.g., critical/extreme), where level 1 is the lowest risk (e.g., low). These levels explain the exact severity of the overall risk of the system model. On the first hand, the advantage of that is highlighting the security weaknesses in the system design, which need more security concerns. On the other hand, it allows focusing on the risks that the user could not accept.

After all potential threats in the system model are detected, the user can generate full documentation on all the details of the analysis process (i.e., threats, likelihood, impact, risk, and the units affected). This process helps collect all system model information as evidence of the system design's status and associated threats that could have a negative impact on system architecture.

Web-based Front-end (Management) This is ThreatGet's second part as a web interface; this interface gives users more flexibility in extending the tool capabilities by handling components, security measures, and risk matrix. A web-based front-end is chosen here to support collaboration, e.g., multiple experts can work on the same knowledge base. It offers a wide variety of vehicular components and security measures. However, some components cannot be defined in ThreatGet, or a specific security measure is undefined. In this case, the user needs to create the required entity (i.e., component or security measure) to use in the system model as required. Therefore, the tool

provides this service to allow the user to create new components and a customized set of security measures to expand the capabilities of ThreatGet in order to be applied in multiple application domains. The user can handle these services through the front-end web user interface, and then the newly created entities are automatically integrated into the Add-in to use in the modeling process.

Also, the likelihood and impact parameter values can be adapted by the user for matching with the user's organization risk scheme. ThreatGet also offers the feature of managing all of these parameters to allow the user to define customized risk levels to fit using in a wide range of system designs. The modified values will be automatically updated and considered in the risk assessment process to match the exact user needs in the risk assessment process.

Web-based Back-end (Threat Analysis) This is the backbone of the ThreatGet, which manages the process of threat analysis. This process comes after the user completes the system model to identify possible system architecture threats. The entire model is uploaded to the "Analysis Engine" on the back-end side with all relevant details. ThreaGet applies a wide range of pre-defined rules to the system model for detecting potential threats. These rules are defined in a specific language developed by AIT and stored in the back-end knowledge base. The analysis engine then uses these rules and all system design information (i.e., components, security measures, connections, etc.) to define the exact threats in the system model. The detected potential threats are then sent to the EA Add-in to display locally.

ThreatGet in an Automotive Example

ThreatGet is developed in order to be incorporated with the early stages of vehicle design and assesses the risk to mitigate it. As a simple example, an infotainment unit receives data from internal interfaces like USB and CD. The infotainment unit is directly connected to the vehicle's main ECU; this ECU has communication channels with another ECU, which regulates vehicle speed and steering. Via the vehicle-to-anything V2X unit, the vehicle has direct interaction with external connection points like another car and RSUs. Figure 2.10 illustrates this scenario using ThreatGet.

This example illustrates various communication channels between internal and external vehicle unit interactions. Attackers always try to exploit security vulnerabilities to initiate malicious actions against the vehicle system. For example, installing malicious software through the vehicle interface such as USB and CD could result in various potential consequences, such as the effect of safe vehicle operation, breach data confidentiality, or other negative consequences. ThreatGet helps to detect such a threat to quickly mitigate it at the early stage of the vehicle life cycle. As shown in Figure 2.11, ThreatGet detects this threat, which is classified as "Tampering," according to the STRIDE category, with a risk level of five, according to the likelihood and impact parameter values. For describing threats, ThreatGet provides a graphical summary of the elements affected by this threat. As illustrated by the figure, the USB and CD could be the way attackers can install malicious software to the Infotainment unit, affecting the entire vehicular network.

ThreatGet detected other 71 threats; these threats are classified according to the STRIDE model. Table 2.2 illustrates the classification rates of all detected threats in this example.

Each threat has a specific risk level based on impact and likelihood values, as illustrated in the pie-chart in Figure 2.11, ThreatGet assesses the overall risk of this example. The chart represents the overall risk evaluation by the tool for this vehicular model. For this model, the tool assesses 33% of the threats classified as level five as the highest level of severity, where 32% of the threats are evaluated as level one of severity, indicating the lowest level of risk. The other evaluated

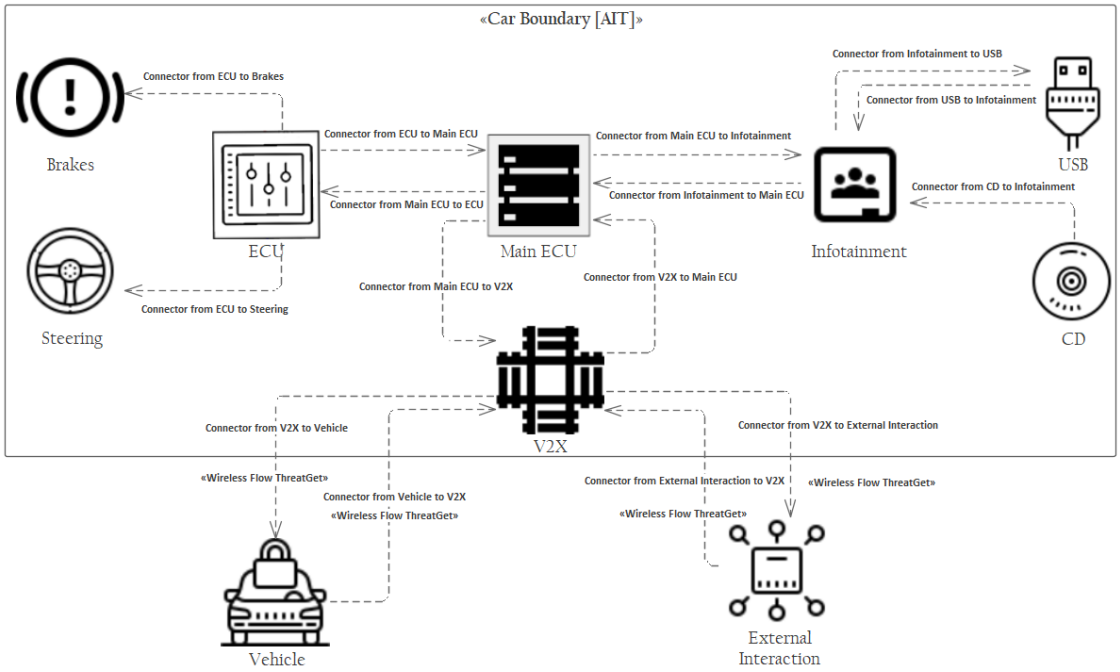


Figure 2.10: Internal and external interactions between different vehicular units

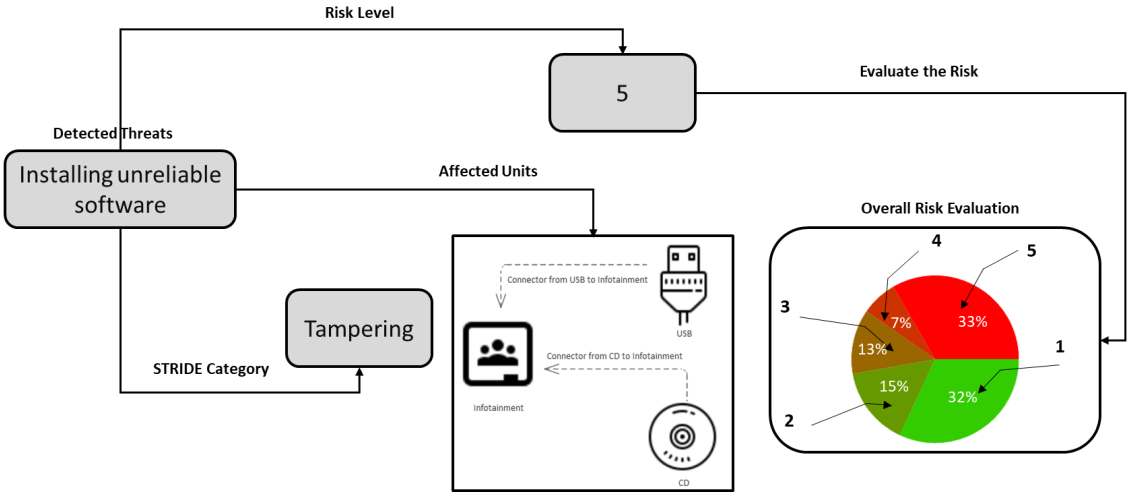


Figure 2.11: The detected threat and related information

risk levels (i.e., two, three, and four) are 15%, 13%, and 7% of the identified overall threats, respectively.

Automotive Cyberattacks

In 2010 the automotive sector started to focus more on cybersecurity when research groups from Washington University and California San Diego demonstrated the ability to control critical components in a vehicle such as a brake system by injecting a code into the CAN-Bus (Controller Area Network). The following year, three different remote attack ways are shown by the same

Table 2.2: The rate of the identified threats according to the STRIDE model

Threat Category (STRIDE)	Number of Threats
Spoofing	**5**
Tampering	**37**
Repudiation	**11**
Information Disclosure	**6**
Denial of Service	**6**
Elevation of Privilege	**7**
Total	***72***

group for executing the code on a vehicle, including the mp3 radio, Bluetooth, and telematics unit, when code has been injected into the CAN bus that could impact any physical unit in a vehicle. The study from remote attacks is noteworthy because it revealed that cars were not only locally vulnerable but also could be attacked from across the world [MV15]. The authors in [MV15] have collected data on many vehicle architectures to classify which vehicles can pose the lowest challenges to an intruder. They found that the 2014 Jeep Cherokee attack looks simpler architecture and several advanced physical features that would be used in their research.

Ref [SDK19] discusses the exiting taxonomies in the automotive development process. It also provides a brief overview of the security vulnerabilities and exiting database for security vulnerabilities. The development of vehicles opens new ways for different attack surfaces that an attacker could seize to reach a malicious goal. Furthermore, researchers check different ways that the attackers could follow to access the internal communication of the vehicle, which influences the vehicle functionalities by unauthorized methods. One possible manipulation of the vehicles' functionality is controlling the window lifters, signal horns, and the lights, which could be how attackers can control specific parts within the vehicle. However, in some other cases, which we could be classified as a worst-cases, these attacks could lead to vehicualr damage or result in personal injuries or death; if the functional safety control units were controlled by attackers. This could include controlling the vehicle acceleration, brakes, or steering the vehicle while driving. Also, the external communication interfaces of the vehicles are considered the other way the attackers could get access to the vehicular assets through wireless communication. Regarding autonomous vehicle's development, study security weaknesses in this kind of vehicles still open research area. Autonomous vehicles are equipped with a wide range of sensors for perceiving information from the environment, such as the camera and light imaging detection and ranging (LiDAR). Researchers demonstrated the capability to blind the camera and deceive LiDAR with fake data. Also, they succeeded in compromising the recognition of traffic signs to imitate the false data in autonomous driving vehicles. These examples show that the variety of potential attack scenarios could take place against the vehicular network [SDK19].

In addition, Ref [Smi16] gives an overview of different classifications of attacks that could take place against the vehicular network. It provides a range of different examples as potential threats could exploit specific security vulnerabilities in the assets of the vehicle, such as installing malicious codes on infotainment unit, jamming key fob signals, installing compromised vehicle software updates, and many more. Ref [GCM] classifies the vehicles into two categories as a malicious vehicle (M-vehicle) and a normal vehicle (N-vehicle). The M-vehicles could be distributed on the highway as a group of malicious vehicles for performing a cooperative attack against the other vehicles (e.g., N-vehicles). The malicious vehicles are able to perform a series

of velocity changes, which could affect the density and velocity profiles of the normal vehicles. This work studies the interactions between these two types of vehicles using the partial differential equations. The study performs analytical expressions to control the velocity input of the malicious vehicles, simulate the results, and confirm the validation of the proposed theory.

Different examples of the cyber-attacks on vehicles such as electromobility shared cars, and Automated Valet Parking (AVP) are discussed in [HM17]. The attack on electromobility shared cars could be initiated at the charging point; the car uses M2M protocol. This protocol is used by the card to authenticate itself, add the charging points, monitor the charging process, report any issues, and complete the charging process. The key point here is the integrations between all the stakeholders such as infrastructure provider, municipality, the energy company, and billing service provider. The identity should be checked for different purposes, such as billing purposes and configuring the charging process. The attacker could try to masquerade as the legitimate entity and disrupt the authentication process in different ways. Therefore, if no strong authentication mechanism is applied, the attacker could gain access to the critical internal infrastructure, which could lead to severe consequences. The second example that has been presented in this work is the attacks on the car-sharing in which the car has a direct connection to the back end server through a GSM connection. The car is continuously sharing data to the back-end server, such as its position and operational data. There are different cyberattack scenarios that could be performed on the sharing cars, such as spoofing the GPS signal, attack on the GSM communication, access the personal data, or denial of service attack. Another attack introduced in this work is Car2Go [16]; the user only opens the car with a special card that has RFID (Radio-frequency Identification). The RFID authenticates itself with the car, and then the car transfers the RFID data to the back-end server via GSM connection of the telematics unit. The car's doors are opened when the authentication process is approved. A four-digit pin is required to verify the user's identity as part of a two-factor authentication using a simple challenge-response authentication, illegitimate users may be able to masquerade as legitimate users and get unauthorized access into the car. Smartphones can perform the authentication process for peering communication with the back-end server to open the doors and control the car, in which multiple attack scenarios are possible. Some cyber-attack threats relevant to the AVP are also presented in this work, mainly regarding the communication between the vehicles and the infrastructure. The work discusses some of these cyberattacks that could impact the AVP functions, such as when an attacker tries to steal or gain unauthorized access into the vehicle. Also, the attacker could directly affect the sensors. The research also mentioned that the parking house management system could use an operating system that is not patched correctly or has expired support, leading to potential threats. Threats also could be propagated through malware spreading through the USB ports or the network that could affect the connected units within the system network [HM17].

2.2.3 Security Requirements Engineering

This chapter has introduced how cyber attacks could be propagated in the vehicle network by different malicious events (e.g., threats) that could be triggered by an attacker to reach a malicious goal by exploiting existing security weaknesses (e.g., vulnerabilities). This next section discusses how to protect the vehicle from different cyber attacks and fill the gaps of the existing security vulnerabilities by selecting the proper security requirements. Ref [FGH+10] introduced a conceptual framework for different methods of security requirement engineering. This work focused on

[16] https://www.car2go.com

security requirements elicitations analysis and the applied proposed conceptual framework for comparing and evaluating the security requirement engineering methods such as Secure Tropos, Common Criteria, SREP, MSRA, and other methods based on the UML and problem frames.

Secure Tropos: is a software development methodology based on the paradigm of agent-oriented software development. It deals with implementation activities in the development process of software, concentrating on the initial stages of the development process [FGH+10]. Such methodology offers a modelling language for security-aware processes and set of other algorithms to support the analysis of the security from the early stage of the development process [OML20]. The models in Tropos are defined in a metamodel consists of different concepts and relationships, as defined in [FGH+10], as follows:

- **Actor:** an entity that has a goal within the system or organization; it could be software or physical.
- **Goal:** this represents the interests of actors toward the system. Tropos classify the goals into two different types; hard goals for describing the conditions that actually need to be achieved by the actor where the second type is the soft goals that are typically defined as a set of non-functional requirements.
- **Plan or Task:** abstraction of something to be done.
- **Resource:** a physical or information entity.
- **Dependency:** actor depends on another entity to achieve a specific goal or deliver a specific resource.

Tropos distinguish between different phases of the development including early requirements, late requirements, architectural design, detailed design, and implementations. Throughout the development process, the model is continuously improved and extended until complete artifacts are defined.

Multilateral Security Requirement Analysis: this aims to apply the principle of multilateral security during the phases of the security requirement engineering process through the development of the system. This can be done by analyzing security and other needs for all the stakeholders. This method is based on multilateral security and viewpoints of oriented requirements engineering. This method is based on seven steps for analyzing the security requirements, as presented and discussed in [FGH+10]; these steps are defined as follows :

- **Identify stakeholders:** all parties that are included within the system interest.
- **Identifier episodes:** identify a set of functionalities that could have an interest to the users.
- **Elaborate security goals:** identify and describe the security goals for different stakeholders.
- **Identify facts and assumptions:** define the properties of the environment relevance for security goals.
- **Refine stakeholders view on episodes:** elaborate the facts, assumptions, and the relationship between episodes of the stakeholders.

- **Reconcile security goals:** identify conflicts between goals and establish a consistent set of security requirements.

- **Reconcile security and functional requirements:** trade functionality for security in case of conflis in the functional ad security requirements.

Security Requirement Engineering Process (SREP): is an iterative security requirements process based on a unified process software life cycle for handling multiple phases. SREP is asset-based and risk-driven, and follows a common criteria, which supports security requirements. It also uses knowledge of assets, threats, and relevant countermeasures. SREP contains nine activities that are described in [FGH+10], as follows:

- **Agree on definition**: defines a set of security definitions, security policies, security visions of systems information; all of these definitions are defined in a security vision document, which is defined as a list contains the most important assets.

- **Identify vulnerable and/or critical assets**: analyse the functional requirements to identify the critical assets.

- **Identify security objectives and dependencies**: once the assets are defined in the previous activity, then the relevant security objectives could be recovered. The security objectives should be refined in a sequence of iterations by defining the dependencies between these objectives, then collect all of these objectives in a security objective documents based on the common criteria assurance classes.

- **Identify threats and develop artifacts**: different methods can be used to identify the potential threats relevant to the previously identified assets such as use cases, misuse cases, and threat attack trees.

- **Risk assessment**: the likelihood, impact, and the relevant risk for each threat are estimated, and then the results can be defined in the risk assessment documents.

- **Elicit security requirements**: identify applicable security requirements that can handle the previously identified threats. Furthermore, the relationships between the security objectives, functional requirements, and threats are used here for this purpose, then the results are collected into the security requirements specification document.

- **Categorise and prioritize requirements**: security requirements are ranked according to the risk evaluation.

- **Requirement inspection**: to validate the selected security requirements, the common criteria assurance is used in this case. If all the security objectives are fulfilled, then the selected security requirements are satisfied, and all of these results are registered in the security requirements rational document.

- **Repository improvement**: the security resources repository can be extended according to the new elements and then defined security targets according to the common criteria.

Common Criteria: This is an international standard aim to achieve the comparability of IT security. The common criteria starts with the description of the functional requirements, system architecture, and describes the working environment. It also includes threat analysis to describe the potential threats then followed by the evaluation of the severity based on the risk analysis. The security requirements are documented in security objectives, which define a set of countermeasures against the identified threats. Also, the security requirements are a refinement of the security objectives, which define as a set of security requirements for the Target of E valuation (TOE) [FGH+10]. Common criteria (ISO 15408) [ISO09a] presents three distinct parts, as discussed in [noa17a], these parts are defined as follows:

- **Part one: introduction and general model**: introduces the common criteria and define the general concept and the principle of the IT security evaluations; the first part can be found here [noa17a].

- **Part two: security functional components**: gives a set of functional components that helps the standards template of functional security requirements for the TOE. It also defines a set of functional components and describes them in classes and families; the second part can be found here [noa17b].

- **Part three security assurance components**: introduces a set of insurance components for support in the assurance required for the TOE. Also defines a set of assurance component is defined in families and classes. This part includes seven assurance levels that are called Evaluation Assurance Levels (EALs); the third part can be found here [noa17c].

The EALs are used for rating the assurance of the TOE. The increase of assurance from level to another could be identified within the same assurance families/classes or other assurance components in other families/classes [noa17c].

The protection profile is discussed in Chapter 4 and Chapter 5, which is used in this thesis to define a set of potential threats related to a particular TOE with relevant assets to define the related security requirements based on the common criteria.

UML-Based Approaches: Ref [FGH+10] discusses some of the approaches based on the unified modeling language (UML) notations such as misuse case, secureUML, and UMLsec.

Misuse cases: this is an extended traditional use case approach to consider the misuse cases that present unwanted behaviour in the the system development process. Misuse cases are suitable for a design system that needs different securities. Also, it includes consideration for all the CIA goals with cooperation of the common risks and threat analysis techniques.

SecureUML: This is a UML based modeling language for developing secure distributed systems. It focuses on embedding rule-based access control policies into the UML class diagram. However, the SecureUML does not focus on the attacker model.

UMLsec: This is a UML-based modeling language for critical systems. This approach focuses on the design of machines in accordance with the three CIA goals.

MORETO As a part of this thesis work, a new security requirement methodology is introduced based on the security standard IEC 62443-part4-2 [IEC19], as discussed and presented in [SKS18a]. The Model-based Security Requirement Management Tool (MORETO) is a tool for security requirements analysis, allocation, and management using SysML/UML models. MORETO is developed by AIT, at the center for Digital Safety and Security. MORETO reuses different features driven by concepts and knowledge of system modeling. It gives the user the flexibility of the modeling process in different SysML diagrams. MORETO is an Enterprise Architect (EA) plugin for managing the IEC 62443 security standard.

EA is a visual modeling software and designing tool based on the UML provided by Sparx Systems [Arc20]. EA gives users flexibility to improve functionality by defining Model Driven Generation (MDG) Technologies. MDG Technologies allow users to extend EA's modeling capabilities. MDG Technologies seamlessly plug into EA to present additional toolboxes, profiles, patterns, templates, and other modelling resources [Sys20]. MORETO is reliable and flexible to model safety & security requirements applied to different components and system architectures.

Figure 2.12 shows the workflow for all MORETO phases. The user can define system components, scan the security vulnerabilities in the system, and cover the detected security weaknesses by suitable security measures defined by IEC 62443 standard.

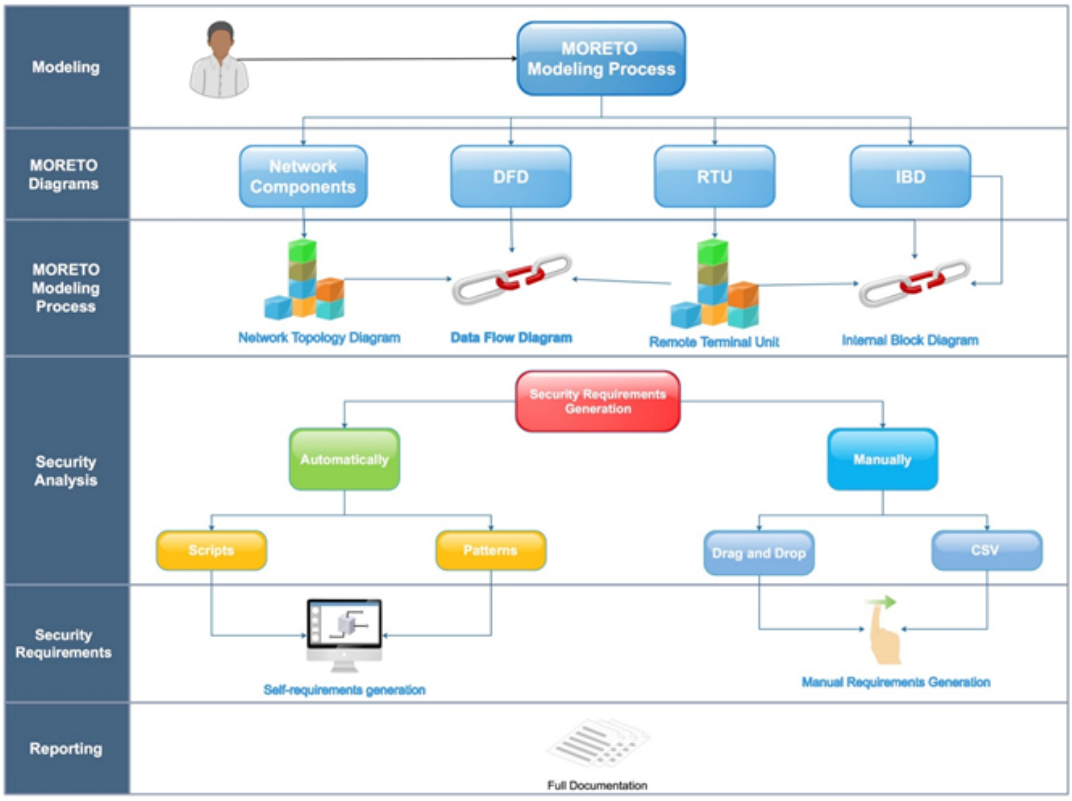


Figure 2.12: Workflow of MORETO Tool

The user chooses the appropriate diagram, which fits the requirements for describing the system components. There are four main diagrams that can be used with the MORETO tool. It can be used to define a complete network topology. The Dataflow Diagram (DFD) can be used to determine the flow of data through components. The Remote-erminal Unit (RTU) diagram has additional

elements to describe an operating plant. Finally, the Internal Block Diagram (IBD) can be used to express the internal structure of the components. The security analysis phase constitutes an integral part of MORETO, which is able to scan all elements of the user's model and detect security flaws and risks. Afterward, MORETO covers these flaws with proper security requirements. Finally, MORETO is collecting all the details of the security analysis and requirements in a final report.

Security Analysis in MORETO: Security requirements generation is one of the unique services provided by the MORETO tool to cover security gaps in the user's model. The security generation process could be manual or automatic as is depicted in figure 2.12 at the "security analysis phase".

Manual Mode: the user uses this mode to create or generate security requirements. This process can be done either by drag and drop or by importing a CSV file.

Drag and Drop: this feature is standard of system modeling software, so the user could specify the security requirements from the toolbox and then drag and drop into the workspace.

Importing CSV files: importing CSV files is another way to generate security requirements quickly without any efforts from the user side. This feature makes the generating process of security requirements much more comfortable than creating these security requirements one-by-one. This mode supports reusing of security requirements list from former projects.

Automatic Mode: as depicted in Figure 4, there are two different ways of generating security requirements automatically: either by using Automation (Scripts) or Patterns.

Automation (Scripts): the automated process in MORETO scans the whole system architecture and all components and detect security gaps and needs. Afterward, MORETO covers the security gaps with security requirements derived from IEC 62443 security standard. With this feature, MORETO automatically generates a list of security requirements on behalf of the user.

Patterns: are pre-defined templates implemented and provided by MORETO. This feature generates a list of security requirements with their relationships to the components.

Security Risk Analysis by MORETO: MORETO tool plays an essential role in generating a list of security requirements for a vast number of different components that are defined in the modeling process. MORETO scans all components within security zones and generates a list of security requirements based on the IEC 62443 standard for each modeled component separately. For example, if we have a system model with some separated zones, these zones have a set of network nodes such as routers, switches, etc. Figure 2.13 illustrates the list of security requirements, which are generated for the Router device.

Afterward, MORETO generates a complete report, explaining the security gaps which are detected by MORETO, and which security requirements have been selected to cover these gaps.

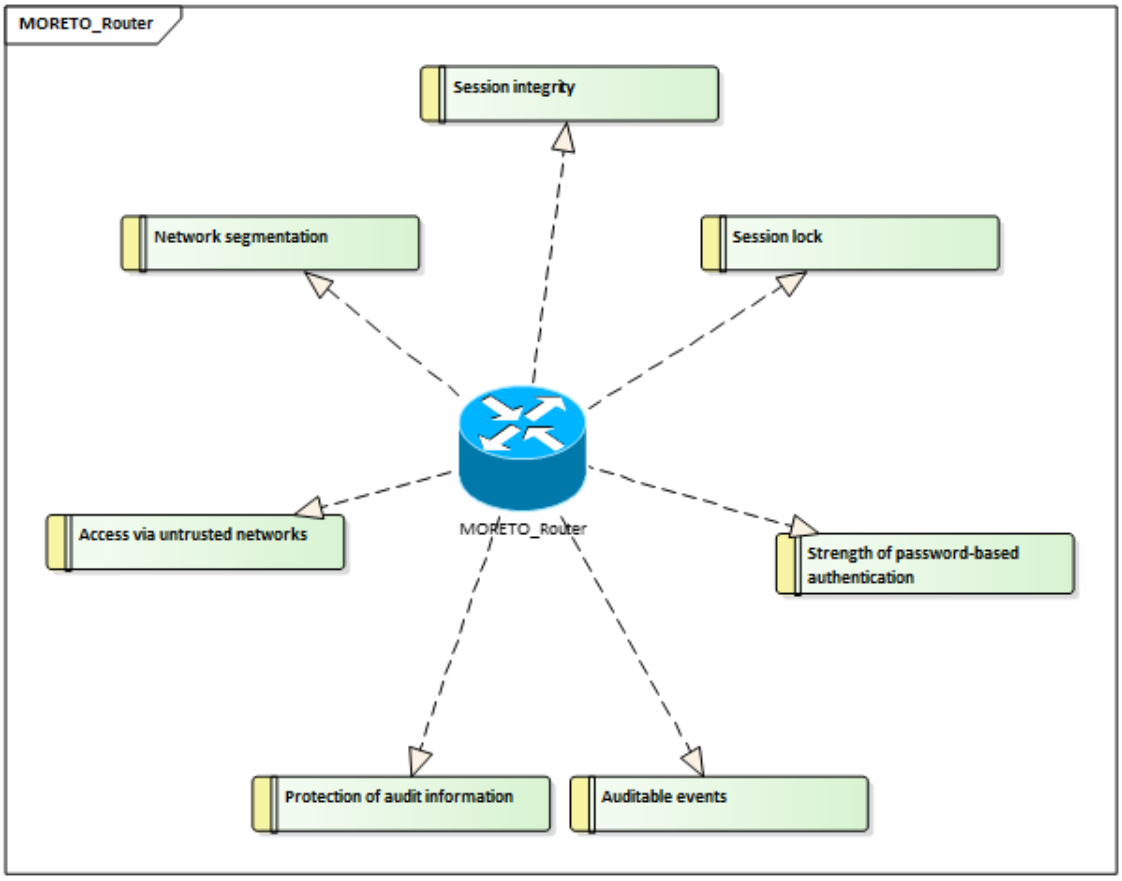


Figure 2.13: List of security requirements of the router component

Example of MORETO This section shows how we can reuse the concept of the zones and conduits on a small scale on the component level (e.eg., router device). Figure 2.14 shows the internal block diagram of the router device.

Figure 2.15, shows how the same concept of the zones and conduits can be applied to the internal components of the router device as a component-level-based model to mitigate the security risk of the router device. Afterward, MORETO can be applied to scan, analyze, and detect security flaws, then generate a list of security requirements for each component separately to cover the detected security gaps of the internal components.

This is especially important for components with mixed-criticality, e.g., where one part is responsible for the outside communication, and the other part is responsible for a timing or safety-critical application. This approach allows us to divide parts with different criticality, group all interactions between them in conduits, and define security requirements based on their needs.

2.2.4 Security Standards

In Ref [WP08], security requirements are defined as the concrete measures needed to satisfy the security objectives such as availability, confidentiality, authenticity, and policy enforcement. This research discusses the security requirement engineering steps based on the vehicle event data recorder (EDR) (i.e., a device installed in some modern cars and trucks to record driving activities

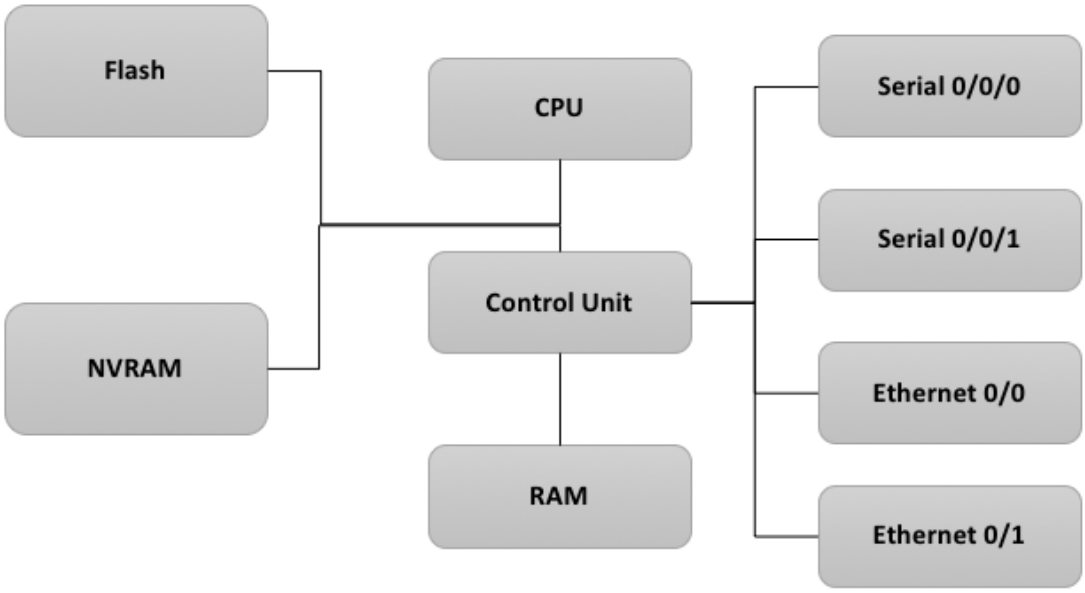


Figure 2.14: The internal design of the router device without zones and conduits

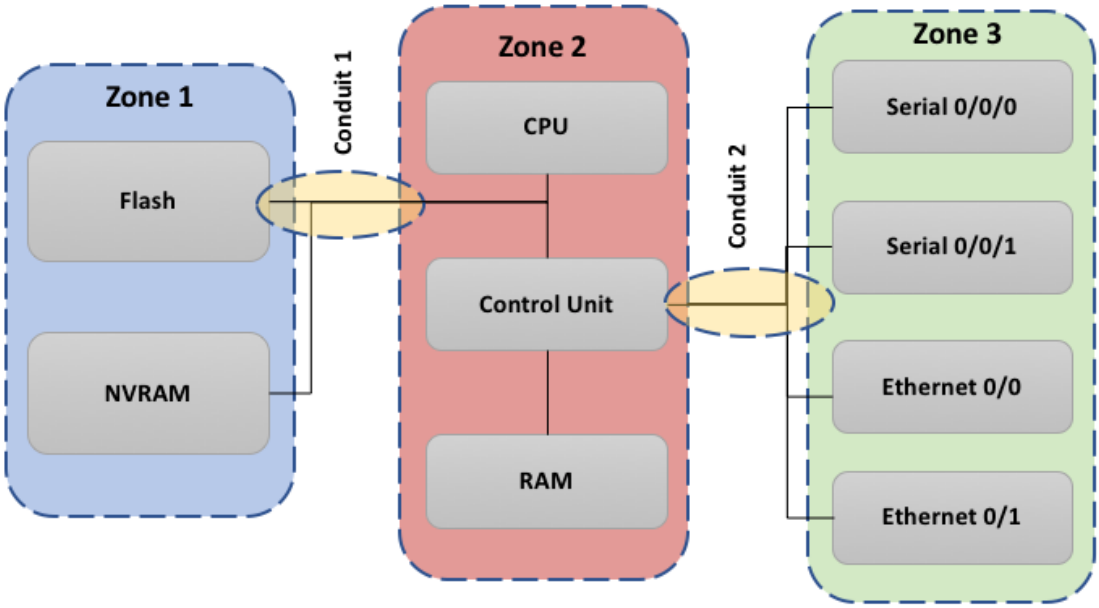


Figure 2.15: The internal design of the router device with zones and conduits

similar to the black box in airplanes clarifying some circumstances of an accident). These steps are defined as follows:

- identify all components in the vehicle that manipulate data covered by security objectives.
- identify effective security assumptions and relevant potential threats.
- drive appropriate security requirements to meet the previously defined security objectives.

Schmittner, Christoph, et al. in [SMR+16] applies J3061 to security engineering for an Electronic Control Unit (ECU) as an example in the automotive domain. This work focused on J3061's seven steps of the concept phase. These seven steps are depicted in Figure 2.16, according to J3061.

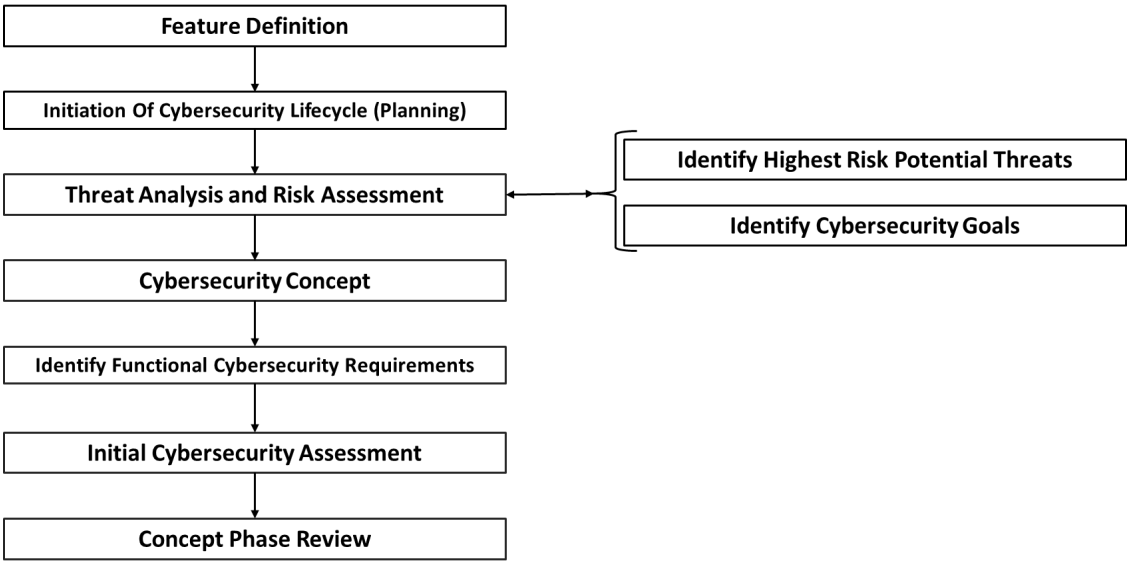


Figure 2.16: The concept phase of J3061

This Figure illustrates seven connected steps of J3061's concept phase; these steps are discussed in [SMR+16], as follows:

- **Future definition**: aims to identify the system's boundaries such as physical and trust to define the main scope of the work.
- **Initiation of cybersecurity lifecycle**: the project is prepared and documented according to the cybersecurity process.
- **Threat analysis and risk assessment**: this step is the main activity of the concept phase in J3061; it contains two sub-activities for identifying potential threats, estimating the relevant risks according to the identified threats and identifying a high-level of cybersecurity requirements.
- **Cybersecurity concept**: includes the high-level of a security strategy that meets the cybersecurity objectives for the previously identified threats.
- **Identify functional cybersecurity requirements**: this step defines the function of cybersecurity requirements based on the high-level strategy.
- **Initial cybersecurity assessment**: aims to estimate the system's security level.
- **Concept phase review**: acts as quality control to review the whole concept phase.

Schmittner, et al. [SMR+16] followed these steps to identify threats and relevant risks to determine high-level security requirements. They evaluated the guidance in J3061 and combined it with their previous experience of the security engineering in relevant domains.

Similar ISO 26262, the ISO/SAE 21434 defines the entire development process in the automotive domain. The standard described the security engineering process in the automotive environment, and it aims to secure the entire life cycle in the vehicle development phases [Til20]. [SGM18] gives an overview of the ISO/SAE 21434 development status. The authors mentioned that the standards address the vehicle's whole life cycle, starting from the developments up to the vehicle's decommissioning. The first draft of the standard was published in February 2020; however, the final standard is expected to be published in early 2021, as mentioned in [Til20].

Ref. [GGV] introduces the first steps towards a framework for automotive information security; the authors give an overview of the existing security standards from relevant domains. This work gives an overview of the existing security standard, such as follows:

- **ISO 27000 family [noa18a]**: the standard contains a series of standards specified in the information security management system to focus on the classical IT systems.
- **IT Grundschutz [noa20a]**: published by the German federal office for information security technology (Bundesamt für Sicherheit in der Informationstechnik) [17]. It aims to standardize the implementations and identification of security measures for IT systems.
- **Common Criteria [ISO09a]**: as discussed earlier in the chapter, the common criteria is based on the assurance of security and aims to meet a particular goal.
- **IEC 62443 family [noa18b]**: this is the family of standards provides a set of procedures for implementing secure industrial automation and control systems.

This work also gives an overview of the standard IEC 62443, which is considered a good example of developing a cybersecurity framework. The IEC62443 standard is used in this thesis to integrate a set of security requirements to address the existing potential threats and fill the gap of the absence of the security standard in the automotive domain to build a secure automotive system.

2.2.5 Security Testing

A vehicle might perform correctly according to the functional requirements; however, it can make other unintended tasks in the process. Furthermore, security tester could miss some of the hidden security defects, which lead to threatening the whole vehicle. Accordingly, the vehicular security requirements must be fulfilled [KN17].

Security testing in a business process could be conducted at the design phase, as mentioned in [LMRM15]. Likely in the vehicular development life-cycle, it is essential to check the applied security mechanism to a vehicle in the early stages of the security engineering process because once the vehicle is built is becomes more difficult to add security. The security verification and validation are the concepts for checking and analyzing the security requirements against the security weaknesses [Som07]. Ref. [KN17] introduced model-based security testing in the automotive industry; it is discussed that the verification process of security requirements is integrated late in the development stages, where both time and budget are restricted.

In the course of our research we introduced the preliminary structure of of the ontology-based security V&V model for the vehicular industry. The model creates a comprehensive ontological representation in terms of classes, subclasses, individuals, annotations, properties, and datatypes of vehicular TOE(s), threats, vulnerabilities, and security requirements (according to CC). A

[17] https://www.bsi.bund.de/EN/TheBSI/thebsi_node.html

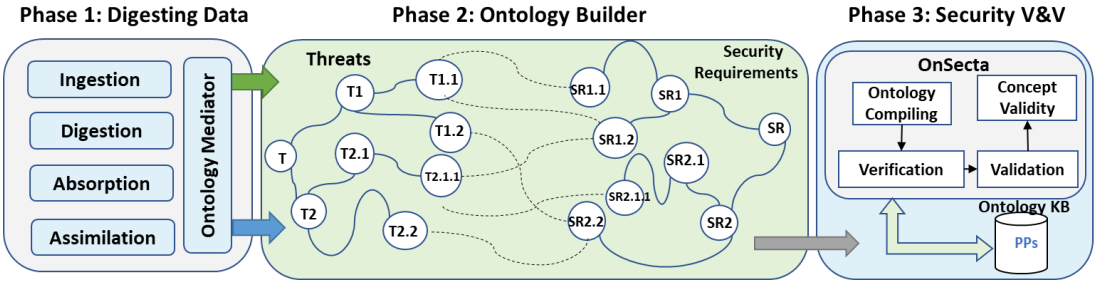


Figure 2.17: The proposed ontology-based V&V model

series of inference rules are applied to the ontology to determine whether or not the selected security requirements cover the security gaps, and confirms if security requirements meet the actual security condition. If this is not the case, it uses a Knowledge Base (KB) of several PPs to select additional security requirements. These additional requirements are applied to handle existing security weaknesses and assure the compliance with protection profiles meet the ST of TOE. The ontologies assist in validating and verifying the operational performance of the security requirements against the vehicular security gaps.

Ontology-Based Security V&V Model

The proposed model uses ontologies to describe a set of representative primitives of classes, individuals, and annotations of security properties of vehicles. The ontology generates new machine-processable meta-data for the vehicle security information, and then the model creates a domain knowledge. The domain knowledge is essential for identifying the relationships between threats and security requirements to verify and validate these security requirements according to CC in one or multiple PP(s). This Section describes the structure of the proposed model, as shown in Figure 2.17. The model consists of three main phases.

Phase One: Digesting Data

This phase receives data of TOE(s) with all related threats and security requirements. This data set is processed by multiple sub-phases to extract the required information [Jos12].

- **Ingestion:** collects the data as follows:
  - list of identified assets with all related information,
  - all the detected threats with all related information details (i.e., name, id, type, description, and risk severity),
  - list of the security requirements according to the selected PP(s).
- **Digestion:** processes the raw data into a standard form that can facilitate extraction of specific values from the original format.
- **Absorption:** extracts all data values which are needed to create an ontological representation from the input.
- **Assimilation:** acts as a filter to get rid of all unnecessary data. For example, the threats with low severity risk are not considered as significant security issues to threaten a vehicle.

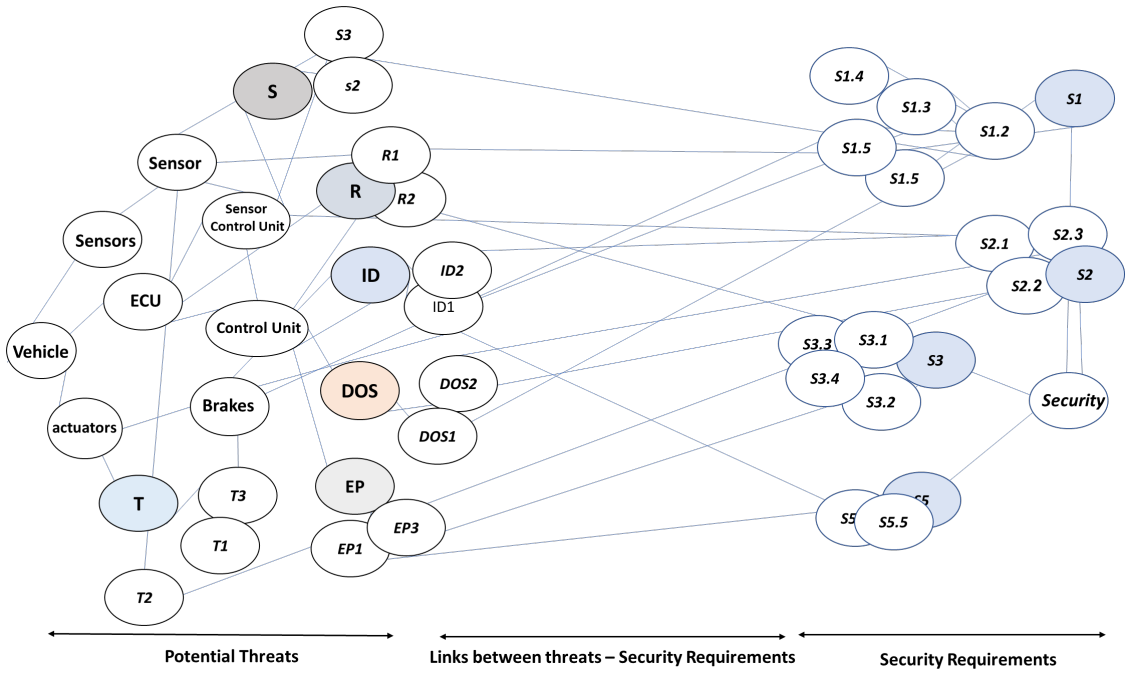


Figure 2.18: Ontology outlook: ontology hierarchy between threats (left) and security requirements (right)

- **Ontology Mediator:** this process propagates semantic annotations or statements (triples) in the form of the subject (threat) – predicate (property) – object (security requirements) which defines the relationships between threats and the related security requirements.

Phase Two: Ontology Builder

This phase generates a comprehensive ontological overview of the threats and its relationships with security requirements. This overview has two main hierarchies:

- **Threats Hierarchy:** this is a hierarchical representation of a typical construction of vehicle threats.
- **Security Requirements Hierarchy:** it is a semantic representation of security requirements that are related to a specific PP for addressing potential vehicle threats.

Afterward, this phase creates an ontology linking between the threats hierarchical nodes and the security requirements nodes. This process defines links between these two hierarchical ontologies, which represent that the selected security requirements can handle one or more potential threat(s). The output of this phase is called "Ontology Outlook" as is illustrated in Figure 2.17 phase two. The left side of the ontology outlook represents the threats, whereas the security requirements are illustrated on the right side. Figure 2.18 shows the structure of the Ontology Outlook; this structure consists of three main parts:

- **Potential Threats (left-side):** this hierarchy has all the vehicular units, which are defined in this case study. The colored nodes define the threat categories regarding the STRIDE model [Sho08]. The leaf nodes represent the actual detected potential threats.

- **The Security-Requirements (right-side):** this hierarchy representing CC are used to handle the potential threats. The colored nodes represent the category of security requirements (i.e., access control, communication port access, use control, data confidentiality, and so on). The leaf units represent the exact security requirements.

- **The Links Between the Two Ontologies:** the links between these two hierarchies can be defined not only between leaves of the hierarchies but also between internal nodes. Accordingly, a node specifying a more general threat type in the threat ontology can link to a subtree in the security requirements hierarchy identifying a set of similar security requirements can fit for handling related security issues [SMM16].

Phase Three: Security Verification and Validation

This phase is the core of the proposed model, which consists of two main parts:

Ontology Knowledge Base: this is a set of specific instances of PPs with all included security requirements and common criteria in an ontology representation format.

Ontology Security Testing Algorithm (OnSecta): is an ontology reasoner uses the Ontology Outlook to perform security V&V procedure:

- **Ontology Compiling:** this process compiles the contents of the Ontology Outlook (i.e., classes, subclasses, terms, annotations, and properties); this allows understanding the ontology linking between threats, and security requirements.

- **Verification:** performs a set of queries for ensuring that the vehicular TOEs are developed regarding CC according to specific PP.

- **Validation:** it assures the compliance of TOEs with PP to meet the actual ST. If that is not specified, OnSecta performs a series of inference rules to select new security requirements from other PPs in the KB to reach the actual ST.

- **Concept Validity:** this activity checks the content of the ontology KB to find new security requirements from other PPs.

2.3 Ontology Mapping

In the doctoral thesis's first steps, we focus on security standards for semantic search engine components. In particular, RDF, OWL access control, XML disclosure prevention, XML/XPath and SPARQL/SPARUL injection attacks, multilevel secure database management system, and semantic web attacks and countermeasures. In this endeavour, we discuss other critical security aspects of semantic engines, such as a denial of service-DOS, man in the middle, session-hijacking, and semantic web services attacks. Then we use N2Query semantic query engine as a case study of this research work.

2.3.1 Security and Privacy Aspects in Semantic Search Engines

Security and privacy are considered one of the most critical issues in the web community. The security concept aims to protect computer systems and information from numerous illegal actions: hacking, intrusion, data disclosure, or damage to information either intentionally or accidentally. Privacy specifies under what conditions information can be exchanged and what are the legitimate uses of these information [KFP+04]. Privacy policies aim to keep information secret in the face of various interaction by semantic web users or agents. Privacy complements information security, granting protection of the individual and non-malicious user [KH15]. Information security includes three main dimensions: confidentiality, availability, and integrity. The concept of information security is concerned with the application security and management set of security measures of a wide range of threats, to achieve business success by minimizing impacts of information security incidents. Information security can be achieved by choosing risk management, policies, processes, procedures, and other security solutions to protect information assets. Furthermore, a systematic and standard approach, such as an information security management system ISMS [ISO14a] should be applied to manage sensitive information so that it remains secure. ISMS is a set of procedures for handling critical data. The target of an ISMS is to mitigate risks and to restrict the impact of security violations. ISMS analyses security requirements for the protection process of information and applies appropriate controls to protect this information. ISMS is based upon risk assessment and the organization's risk acceptance levels designed to treat and manage risks effectively [ISO14a].

ISMS has a big family of standards helping organizations keep information assets secure. ISO/IEC 270xx [ISO14a] is a family of standards that help small, medium and large businesses in any sector keep information assets secure, such as financial information, intellectual property, employee details or information entrusted by third parties. The semantic web consists of numerous types of data, which need more confidentiality and privacy. Various attacks can disclose the sensitive information of the semantic web to malicious users and their agents. This work focuses on the most critical security aspects which threats semantic search engines and its data.

Security Viewpoint

A security viewpoint in the design of a standardized semantic search engine reference architecture model has the benefit of dividing the model into components, and to align various architectural artifacts in order to facilitate security-related activities from analysis to validation, even to support security management in operation.

Figure 2.19 introduces the main architecture layers of our proposed model of semantic query engines. The model aims to map each element of these layers to its security aspects in the vertical model. The arrows shown in this figure represent the mapping process between security requirements, which should be used in the main consideration of the implementation process of the layers of semantic query engines to cover the security gaps. For example, the semantic web layer has some security facets which should be covered, such as access control, and encryption.

To establish the security viewpoint, we propose to include all security topics in one place and relate each topic to the elemental components in the layers of the semantic query engine. In other words, the security aspects in semantic search engines are modeled as a three dimensional layered architecture plus a vertical plane perpendicular to the layers of the semantic search engines. Figure 2.20 depicts a "security viewpoint" to collectively refer to security related issues and the references of the security-related information of the architectural artifacts.

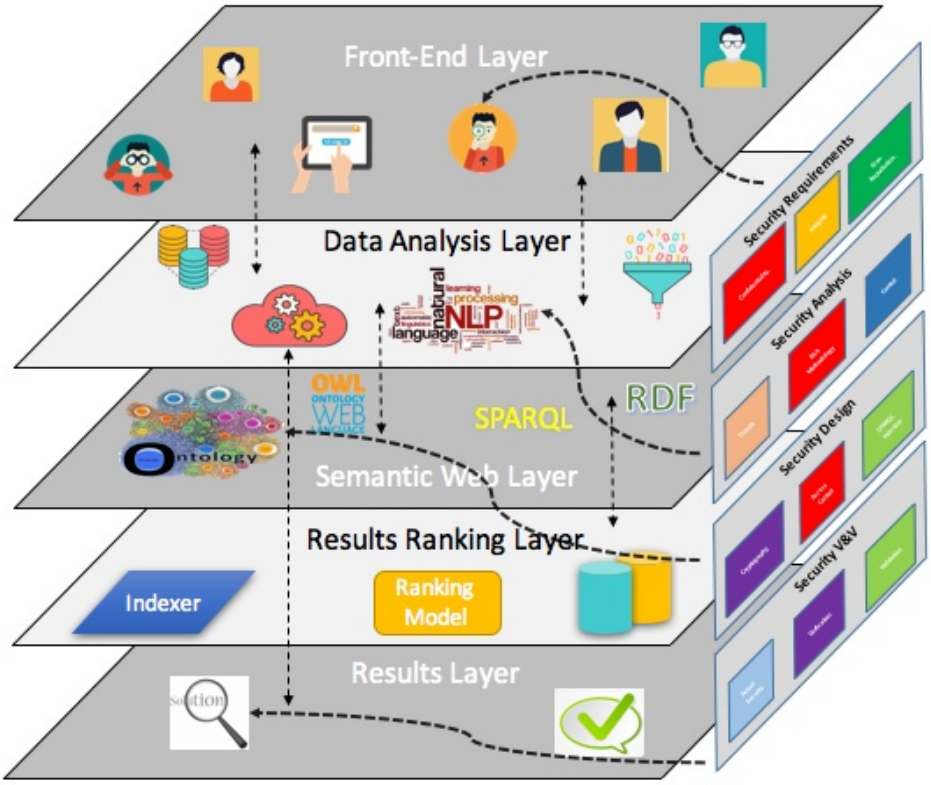


Figure 2.19: Security viewpoint as a vertical plane to the main semantic query engine architecture layers

This design has four main strategies; security requirements, security analysis, security design, and security verification and validation. The aim of that design is to map each element of the semantic query engine with one or more security requirement(s). Then the security analysis process applied to provide solution(s) of security gaps. At the end, the security verification and validation is presented to emphasize the security actual needs. The security requirement level describes the behavior of the functional and non-functional requirements, to cover the security aspects of different components which are provided in the proposed model. Security design, describes the security aspects need for the semantic search engine to define and describe the security methodology of the components of the semantic search engine to deliver a secure and dependable architecture model of semantic engines. Finally, the security verification and validation is carried out to confirm that the operational performance of security requirements are viable. Security verification is applied for determining whether or not a product covers all security gaps in the proposed model. The security validation checks if all security requirements are meet the actual need.

The architecture layers of the semantic search engine in figure 2.19 reflect the interaction between layers of the semantic search engine, starting from firing queries by the user through out the interface at the Front-End layer, to the point the search engine replies to the user with the answers of his\her query. The suggested structure also illustrates how the vertical plane of the security viewpoint can address these issues by creating links between these security gaps and

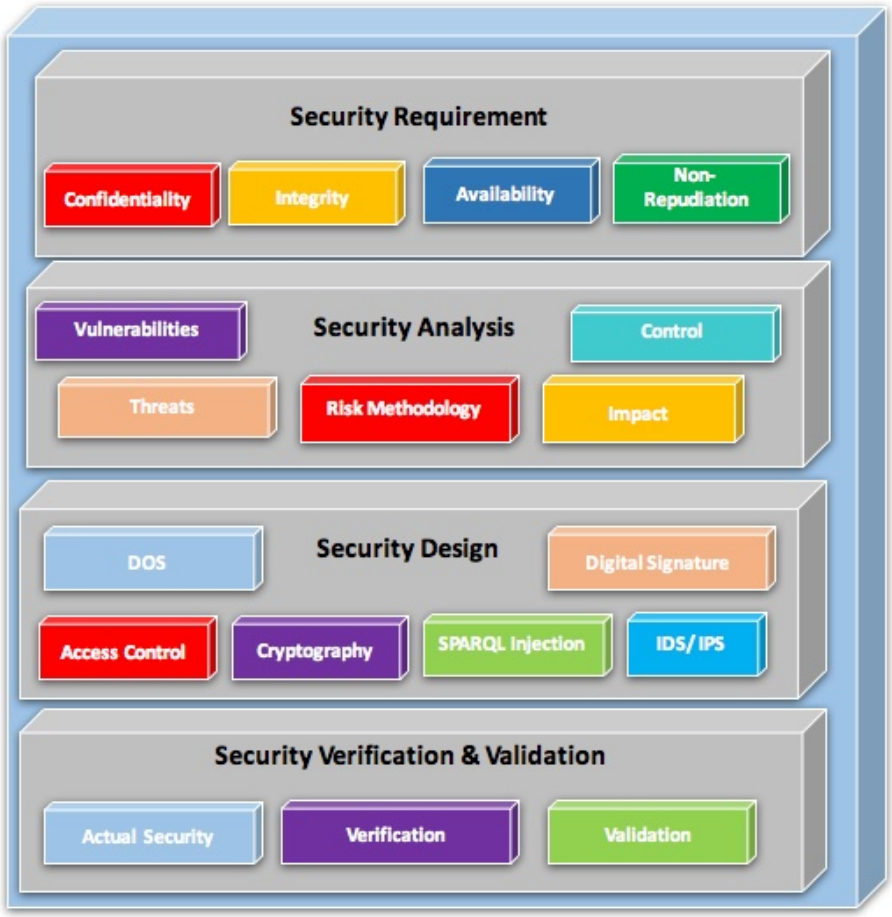


Figure 2.20: Security viewpoint

relevant security requirements. The Data Analysis layer accepts that query and starts to analyse and recognize the semantics of that query by performing text processing methodologies using natural language processing algorithms. Then, the Semantic Web layer receives the query and starts finding a solution(s) to the user's question using mapping techniques by executing SPARQL queries on RDF data. Before sending results to the user, the Ranking layer filters these results for giving the best solution(s) to that question.

Security Issues and Possible Countermeasures in a Semantic Web Context

This section discusses common security issues which can jeopardize the semantic search engine by different types of attacks. Figure 2.21 shows the architecture of the semantic web stack by W3C [18] [noa00]. The model is structured by different components, each one of them has its own characteristics and different security issues to be solved. The following sections discusses briefly security disparities and some of the suggested solutions to overcome these security problems.

[18] https://www.w3.org

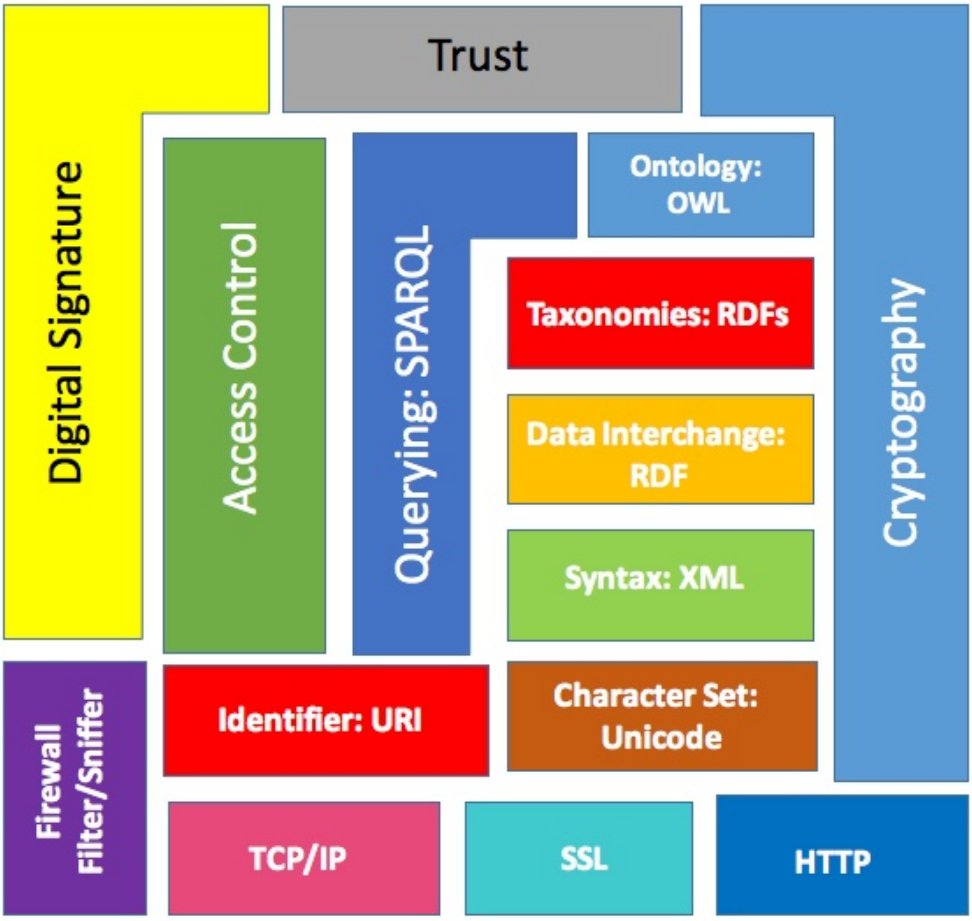


Figure 2.21: The architecture of the semantic web, based on the semantic web stack by W3C

TCP/IP: There is a list of security attacks which threaten TCP/IP and SSL such as Denial of Service (DoS), packet sniffing, TCP sequence attack and so on. Attackers try disrupting the data transmission by sending multiple of fake requests using false IP addresses that affect the performance of servers leading to lost connection between clients and server. Also, there are many kinds of TCP connection threats such as sniffing attack that threaten intercepts and analyze network traffic in the network communication channel [KK14]. The countermeasures to cover the security gaps of these attacks is to apply a filtering process to all incoming traffic to the server. In addition, MD5 and Kerberos can be used as a cryptographic hash function to apply authentication. IP scanning mechanism, proxy firewall, and Stateful Packet Inspection (SPI) can be applied to distinguish legitimate packets for different types of connections.

HTTP: Man in the middle, Session-hijacking, HTTP-GET attacks are the most common threats of the HTTP attacks in the semantic web. The attacker can use SOAP messages to steal information [KPSD10], where Session-hijacking attacks aim to gain authorized access by stealing the session-key. Actually, there are two different types of the man in the middle attack, passive and active attacks. By the passive attacks the intruder sniffs data in the channel without modifying

the message content. By the active one modifies the message contents in transit or deletes it. The encryption of the communication channel is one of the best solutions to solve the problem of the session-hijacking problem. Also, using Hypertext Transport Protocol Secure (HTTPS) is the main security requirement to overcome the network eavesdropping [KK14].

Authentication: Authentication is one of the most common security features proving the identity to another entity, such as a username and password. Authentication attacks have different types of threats as Network eavesdropping, Dictionary attacks, Credential theft, Man-in-the-middle and others. Keeping trust in information is the aim of authentication. Thus, authentication is considered a sub function of trust. The relationship between authentication and trust is similar to the relation between encryption and secrecy. Authentication can be guaranteed in the semantic web by applying signing and verification in XML and RDF layers [Shi03].

Access Control: Access control restricts actions by users or services (typically referred to as the subject) which can be performed on other resources as objects. Over the last decades, there are several types of access control models, including Role-Based Access Control (RBAC), Discretionary Access Control (DAC), Mandatory Access Control (MAC), and Attribute-Based Access Control (ABAC). Entity-Based Access Control (EBAC), as one of the most recent kinds of the access control types. It introduces entities as a primary concept and takes into account both attributes and relationships to evaluate policies. EBAC increases the expressiveness of policies which fit the application domain well [BDLJ15]. The concept of access control is highly recommended in the semantic web environment to increase security and safety of the semantic web assets.

Discretionary Access Control: The concept of Discretionary Access Control (DAC) gives access permission to the user based on the user's identity. DAC has two main parts, which are access rules and access attributes. The access attributes allow the system to set several distinct levels of authorization, where the access rules provide conditions of unauthorized access to access critical information [PJP12].

Mandatory Access Control: Mandatory access control (MAC) is based on the access of objects to several subjects. The subject can be accessed to a particular object by satisfying the relationship between them. Security level helps MAC to define the current access status of an object. MAC is applied to protect networks and block services for users that have no authorize access into networks [PJP12].

Role-Base Access Control: In Role-Base Access Control (RBAC) access decisions are based on the individual's roles and responsibilities within the web. RBAC identifies the roles as a set of objects or actions associated with the subject. It formulates the user's access to the system based on the activities that the user has been executed. The roles are assigned based on the least privilege for the particular object, so this will minimize the damage of information by intruders [PJP12].

Entity-Base Access Control: Entity-Base Access Control (EBAC) is a new version of the access control model based on the comparison of attribute values and the relationships between different entities. Entities can be either attributes or relationships or even both. The concept of entities is introduced in EBAC as a first-class citizen in access control policies. EBAC policies can

navigate through relationships and analyse the properties of associated entities. The comparisons can be composed into complex expressions using logical connectives or predicate logic functions that can reason over relationships [BDLJ15].

XML: There are four types of XML attacks, such as XML injection attacks, XPath attacks, XML parsing attacks, and XML Denial of Service attacks. With the injection attack, the attacker passes a malicious XML code into the stream of XML. The XPath injection is quite similar to the XML injection, the attackers construct an XPath query and send that query intentionally into the victim machine. Here, the attacker is able to access data which may not be accessed, or even able to change his privilege on the victim machine [SS05]. The World Wide Web Consortium (W3C) [W3C] sets security standards to cover the security gaps of XML such as XML encryption, XML digital signature and access control [AA14]. XML encryption is used to encrypt SOAP messages and guarantees the message's confidentiality over public channels. In addition, XML digital signature is used to provide integrity and authenticity of XML documents [Thu15]. XML parsing attacks focus on two types of XML parsing in XML: the first one is the Simple API for XML (SAX) and the second type is the Document Object Model (DOM). The SAX algorithm is parsing XML documents and provides a mechanism for the contents of XML data. Where the DOM parser defines algorithm interfaces to enable programs to access and interact with XML documents. The attacks of these XML parsers can be done by sending malicious XML code to the XML documents but is extremely complex to be detected. The simple way to protect XML documents from the parsing attacks is to use Document Type Definition (DTD). DTD is applied by parsers to validate the structure of the XML documents.XML Denial of Service Attack (DDoS attack) is another type of XML attack, which intends to crash or hang a program using either XML bombs or external entity attacks [KK14].

Message Encryption: Encryption is the process of ciphering messages or information in such a way that no one can read the content of it except authorized parties. By using an appropriate encryption algorithm, the web services messages (SOAP, XML, and semantic data) are fully protected. The confidentiality can be achieved by encrypting the content of messages to keep it secret from unauthorized disclosure [Cha10].

Message Signing: Message signing is used to protect the integrity of messages in transit over the network and to provide proof of the original sender. Signing does not protect the confidentiality of the data, but only its integrity and confidence in the original sender. Message signing can be provided by either message security or transport security. Message security signs each message individually, while transport security protects the entire communication channel (e.g. with SSL) [Cha10].

RDF: The Resource Description Framework (RDF) is a standard method of data exchange over the Internet. RDF is a part of the family of W3C used to describe web resources in a standardized way using rules, notations, and formats. RDF has three main components resources (subjects), properties (objects), and statements (predicates). Anything on the web can be defined as a resource, where the property can describe some other resources. The statements in RDF combine resources with properties' values [Thu15].

The semantic data has to be protected against data disclosure and unauthorized access by user or services. Consequently, the semantic data's storing process is one of the main security

requirements to keep data safe and validated. The most widespread RDF database management system RDF4J (formerly known as Sesame) [SNP+16] is a Java open-source framework for RDF data. The framework includes parsing, storing, inferencing and querying the stored data, and defines access permission to the whole database only. Therefore, a fine-grained solution is required [RDF]. Oracle cooperation solved this problem by defining a new feature called Virtual Private Database (VPD). VPD is a row-level protection mechanism that limits access to specific rows in a relational database or view using a security policy and an application context. The security administrator can use VPD for RDF data to define restricted policies to users or services to access RDF triples involving instances of a specific RDF class or property [Fin17]. Also, Oracle provides Oracle Label Security (OLS) for RDF data which allows sensitivity labels to be associated with individual triples stored in an RDF model. OLS is a data classification to strengthen access controls by restricting users to only access data they have permission to access [Fin17]. Another essential security issue of database security is to keep the stored RDF data secure against internal or external attacks. This led to the development of CryptDB [PRZB11], which is a system that provides practical and confidential methods against data attacks. CryptDB works by executing SQL queries over encrypted data utilizing a set of effective SQL-aware encryption schemes. During the translation of SPARQL queries to SQL queries, a relational database management system can be used as a storage medium for storing RDF data and can be managed and controlled by the semantic query engine by taking all security aspects.

SPARQL Query Language: SPARQL is a query language for RDF data. It is quite similar to Structure Query Language (SQL) used to retrieve data from a relational database. Similarly, SPARQL query language is used to navigate RDF documents and retrieve data matching the query statement. Acquiring unauthorized data is an important security issue which threatens the data integrity on the web. Different malicious actions can do this. The most common one is the injection. The semantic data has two types of injection attacks: SPARQL injection, and Blind SPARQL injection. The injection can be done from a web form into the database logic of an application to acquire unauthorized data access to make unauthorized deletions or alter the data. Blind SPARQL injection and SPARQL are quite similar, except that with a blind injection, an attacker attempts to get data illegally by asking true and false questions through SPARQL statements [AAL16]. Many semantic web systems use a relational database to store RDF triples (subject, predicate, and object) and use a translator module to translate SPARQL queries to SQL. The injection attacks are therefore classified as either SPARQL oriented or SQL-oriented. There are some proposed methodologies to detect these attacks. The most common one is the parse tree [YCZZ11].

Parse Tree Validation: Parse tree is a data structure using for representing the syntactic structure of a statement. It uses a formal grammar for describing the syntactic structure of a statement. A SPARQL parse tree is proposed to detect injection attack, based on a validation technique, to check the syntactic structure against the grammar. Additionally, 'SQL-injection' or 'SPARQL-injection' is also used for auto-fix query [TW07]. The detection process can be done by comparing the parse tree nodes in the corresponding places with the path and value of the user input. For instance, the following example shows a simple query in SPARQL language, and figure 2.22 depicts the parsing tree of that code, are defined as follows.

```
1 PREFIX foaf:  <http://www.emp.com/>
2 SELECT ?name
```

```
WHERE {
    ?name   emp:fname \""+get_fname+"\".
    ?name  emp:lname  \""+get_lname+"\".
}
```

Listing 2.1: A simple query in SPARQL language

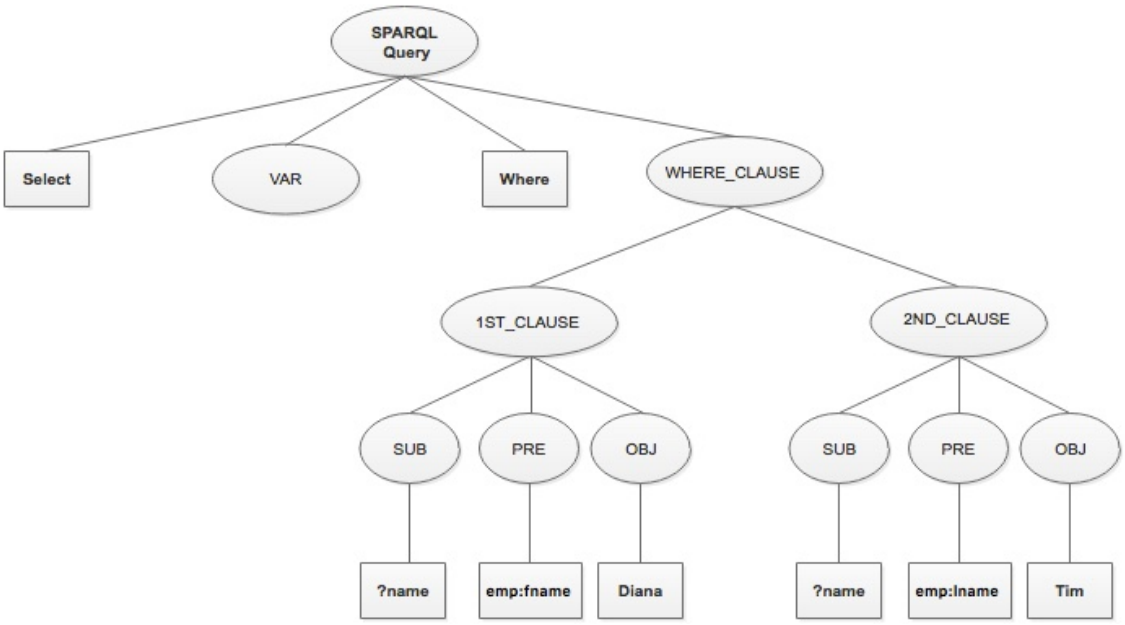


Figure 2.22: SPARQL parse tree

After that code is executed, the lname node with the value Tim in figure 2.22 cannot be found in the result of the parsing process and considers this an attack.

For the SQL injection attack, the sequence diagram in figure 2.23 shows the handling process when an attack happens. The SPARQL query has to be checked by a validation process that examines the SPARQL query structure and gives two possibilities, either attack detected or not. After the SPARQL statement's validation is confirmed, the second step is translating SPARQL to SQL and the validation process checks for an attack. When an attack is detected, there are two options for handling that exception: rejection or auto fixing [YCZZ11].

2.3.2 A Security Model for Semantic Query Engines

This section proposes a workflow and security requirements perspective of our proposed model for semantic search engines. The model is built on two basic requirements:

- The design should be based on a standard modeling language.
- The model should be a simple design to make the traceability of the interaction among system's components possible. So, we decided to depict that design in two separate layers.

Following these requirements, we need to choose the best methodologies to implement a comprehensive and straightforward diagram that reflects our target for realising a semantic query

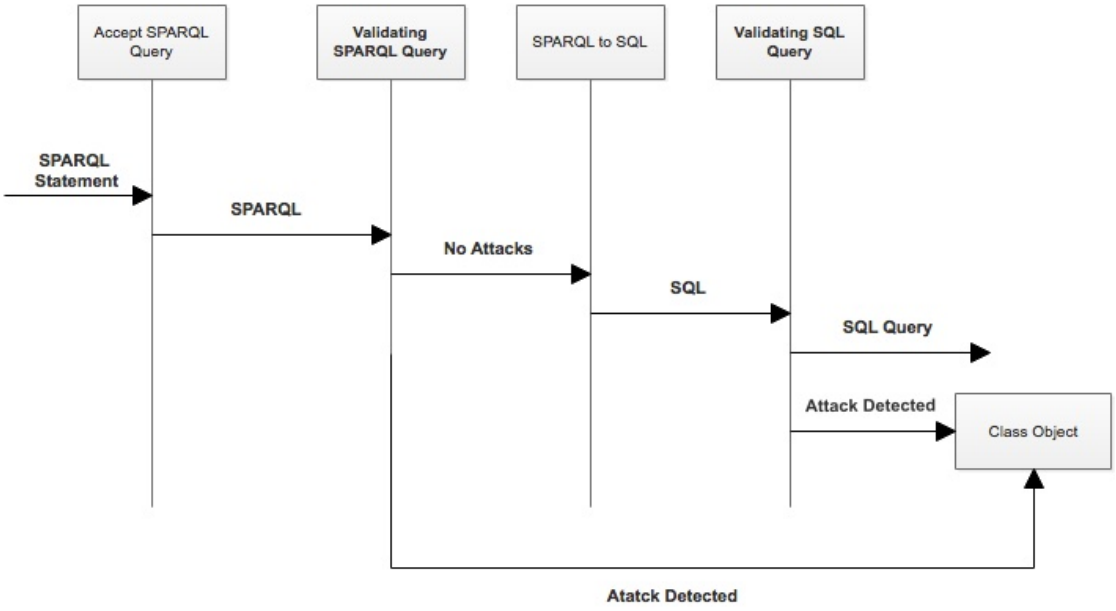


Figure 2.23: Workflow of attack detection process in SPARQL query

engine model. So, Sparx Enterprise Architect [Spa17] was chosen as a modelling tool for the implementation process. Enterprise Architect provides the basis for the modelling process and can cover all aspects of the development lifecycle, providing the process traceability starting from the initial design, development, maintenance, test and project management cycle [MHSP17]. Enterprise Architect can define models in different layers and connect components internally to show the model's baseline view plus other perspectives. Consequently, we designed our model on two levels: the first level shows the model's workflow diagram. The second layer shows the security requirements based on ISO/IEC 27000 information security management covering the security gaps proposed in section 2.3.1. Figure 2.24 depicts the structured layers of our proposed architecture.

The first layer shows the flow control of the main components of the semantic search engine. The second layer defines the security requirements of all components defined in the first layer to cover all security aspects that threaten the data, resource, services, components and other elements of the semantic search engine.

Workflow Diagram

This section defines the semantic query engine's main components and shows a workflow diagram that represents the internal interaction among these components, as depicted in Figure 2.25. The query engine has several components; each has a specific task for delivering back an appropriate answer to the user's question.

User Interface The user interface (UI) service accepts user queries, retrieves results to user queries, and displays ranked results via the same interface.

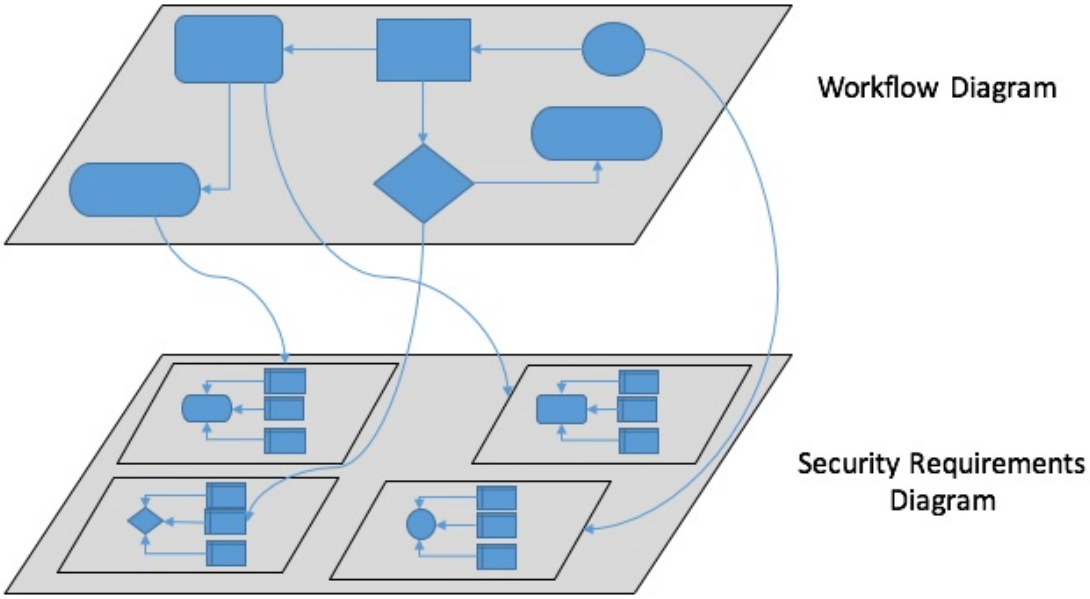


Figure 2.24: Workflow and security requirements perspective of the SQE proposed model

Natural Language Processing This natural language processing (NLP) service aims to understand a human query using advanced natural language processing methodologies to get semantic representation of the user query. That type of processing includes sentence segmentation, part-of-speech tagging, in-depth analysis, and parsing. The NLP service applies five steps to understand the meaning of statements. [SMH+17]:

- **Lexical Analysis** divides the chunk of text into paragraphs, sentences, and words.
- **Syntactic Analysis (Parsing)** checks the grammar of sentences and arranges the words in a manner showing the relationship between words.
- **Semantic Analysis** induces the exact meaning or the dictionary meaning from the text. It is done by mapping syntactic structures and objects into the task domain.
- **Discourse Integration** determines the meaning of any sentence depending on the meaning of the preceding sentence.
- **Pragmatic Analysis** derives the aspects of language based on real-world knowledge.

RDF Crawlers The RDF crawler creates queries by enumerating keywords generated by NLP and sends queries to RDF storage to generate RDF documents. The RDF crawler also can be

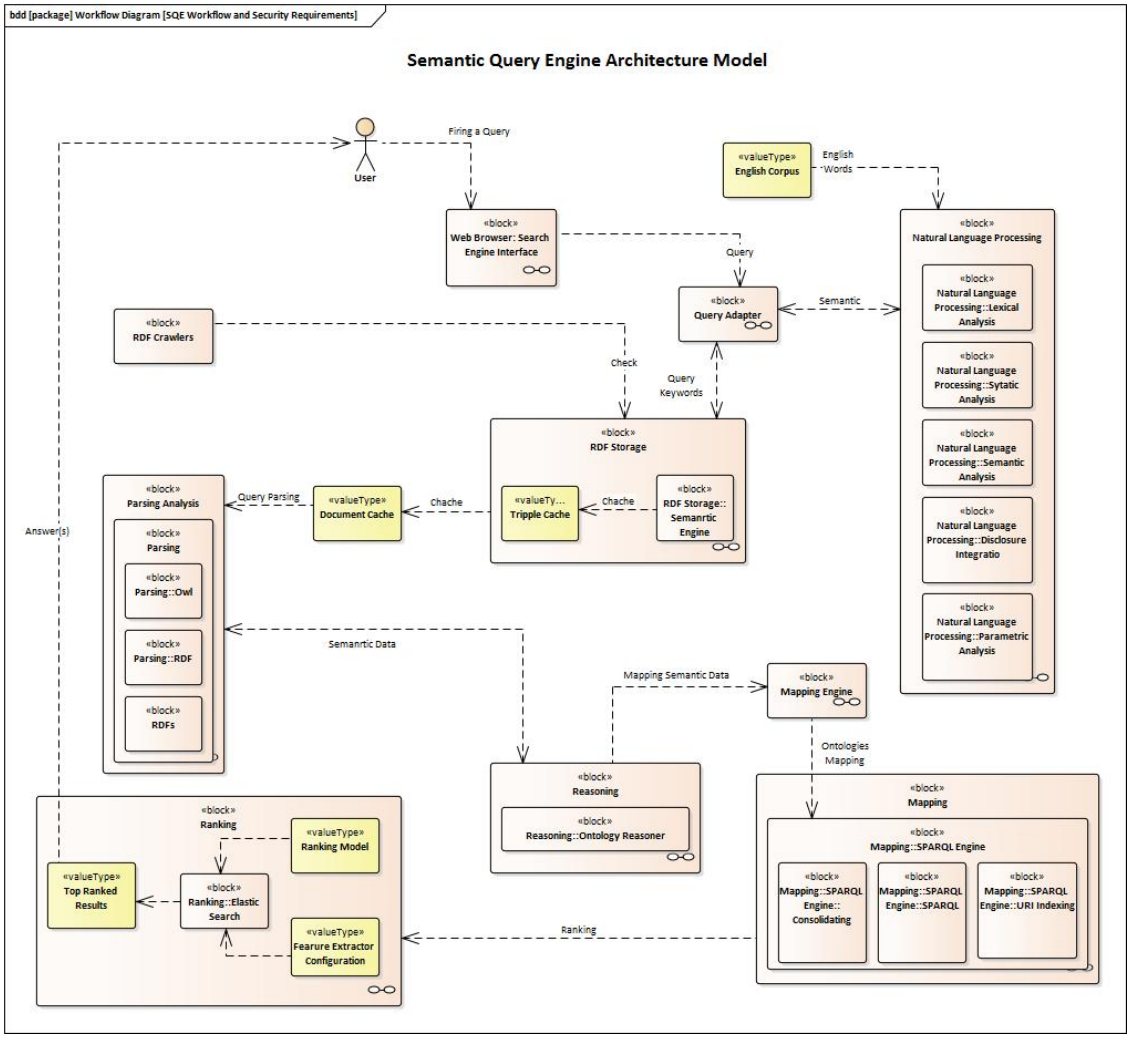


Figure 2.25: The main Components of the semantic query engine

customized to interact and download RDF data from other resources such as Dbpedia, Hannover, DBLP, etc.

RDF Storage The RDF storage is a database for storing and retrieving data triplets. The semantic engine uses the stored data to create and maintain files and send them to the cache then.

Parsing The parsing unit aims to parse and analyze the semantic data and extracts elements and keywords inside this data. As defined previously, the semantic web has structured data representation levels; so RDF, RDFs, and OWL parsers are used in this process to classify objects and subjects from the given statement.

Reasoning The reasoner engine can infer logical consequences from the set of semantic web data described and generated at the previous step.

Mapping The mapping process uses a semantic query language like SPARQL language to map or match between different results.

Ranking The ranking process acts as a filtering task and ranks solutions. Thus, it publishes a list of filtered solutions to the user.

Security Requirements Diagram

This section shows the security requirements of the selected units of the semantic search engine. ISO/IEC 27018 [ISO14b] is the international standard used in this work to define the most common security standards of each element in the semantic search engine. The standard specifies guidelines based on ISO/IEC 27002 [ISO13] taking into consideration the regularity requirements for the protection of Personally Identifiable Information (PII), which might be applicable in the context of the information security risk environment(s) of the (public cloud) services [ISO14b]. Figure 2.26 depicts the security requirements of the RDF Storage unit according to ISO/IEC 27018 standard. The SysML requirement diagram used in this implementation defines our proposed design's internal levels to show the security requirements for each basic element of the semantic search engine as described in figure 2.25.

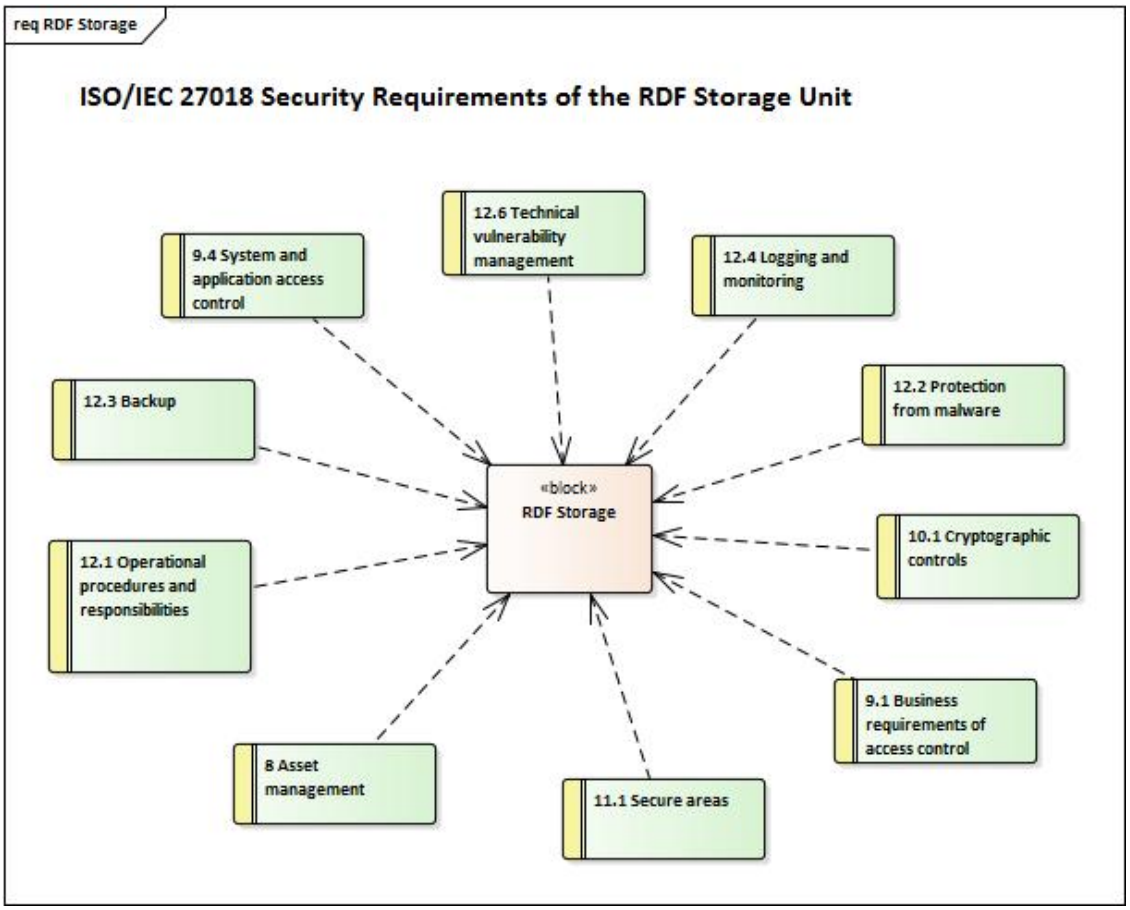


Figure 2.26: Security requirements of RDF storage unit

As illustrated in figure 2.26, there is a list of security requirements for the RDF Storage, which includes data backup, technical vulnerabilities management, cryptography controls, and other applicable security standards.

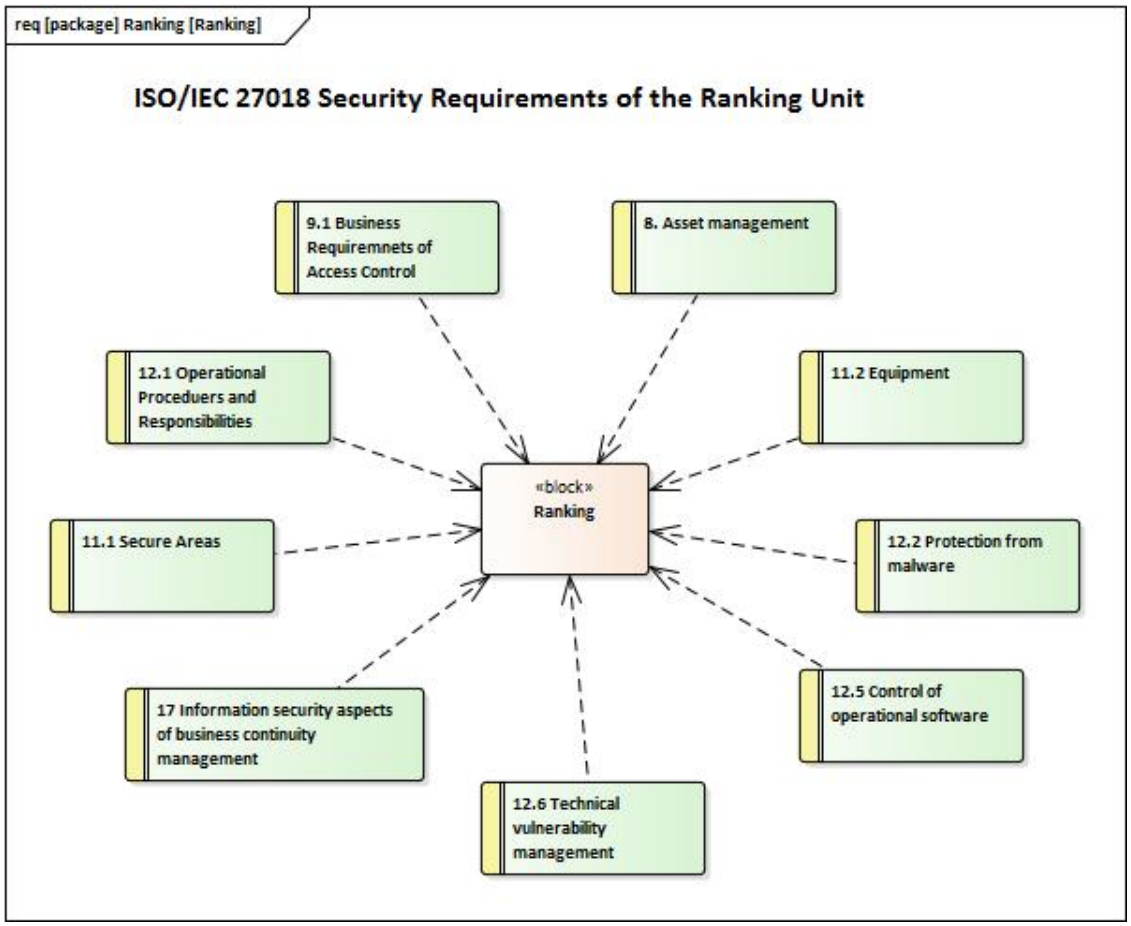


Figure 2.27: Security standard of the ranking unit

In addition, figure 2.27 shows the security standard of the ranking unit. Protection from malware, operational procedures and responsibilities, and other suitable security solutions cover security gaps for this unit.

2.3.3 Case-Study: N2Query

In our research, we designed and developed N2Query, an implementation of our generic semantic query approach as a component of the N2Sky infrastructure. N2Query is a semantic query engine based on N2Sky [SM13] as a virtual organisation for the computational intelligence community. N2Sky enables a virtual cooperative organisation for the computational intelligence community. It provides access to the neural network-specific knowledge which is distributed over the internet fostering multi-cloud resources. Due to the architecture of N2Sky, the search for specific neural network resources providing solutions to given problems can be a very time consuming and challenging task. Thus, we developed N2Query [SMM16, SMH+17], an implementation of our semantic query engine model as a component of the N2Sky infrastructure. N2Query enables users

to specify their queries in natural language description of the problem statement and delivers a list of ranked N2Sky resource - URIs of trained neural network objects, which provide solutions to these problems.

The main components of the N2Query architecture are [SMH+17]:

- A semantic querying interface allows the user to specify his/her problem description in natural language form.
- As the first step, an ontology mapping mechanism recognises the semantics of the natural language query using the problem ontology. The second step matches the N2Query's problem ontology against, already constructed, with solution ontology of available neural networks resources solving the given problem.
- A ranking mechanism delivers a list of links to neural network resources of the N2Sky infrastructure.
- An XML based domain specific Neural Network resource description language to maintain the problem and solution ontologies.

2.3.4 N2Query: Semantic Query Engine

N2Query aims to solve different problems by firing queries, and the results are displayed in the same interface. Our approach consists of two main phases, the ontology integration phase, and the ontology query phase [SMH+17].

This research aims to introduce a semantic management framework for neural network resources. The framework is a new technology of problem-solving techniques using the N2Query system by allowing users to describe problems in natural language description. N2Query delivers the solution to fostering semantic techniques. We have outlined the implementation process of problem-solution ontologies using semantic web languages and other tools. This proposed concept aims to provide solutions to given problems can be a time consuming and challenging task. This concept was based on the ontology approach for mapping between solution ontology and the problem ontology for a wide range of neural networks that are problem-based.

2.3.5 N2Query Architecture

By using N2Query, a user can send queries in natural language via the N2Query interface and communicate with the N2Sky infrastructure to find suitable solutions. Figure 2.28 shows the high-level process of N2Query. The detail of every stage through the process will be covered in the subsequent sections.

The user's query in natural language is process by the Stanford Language Processor [19]. The processed problem statement is then sought out through the already built problem ontology. In order to generate the ontology mapping for the problem, relevant solution terms are linked with nodes in the solution ontology. The ontology mapping results in the identification of the neural networks aim to solve the submitted problem. The retrieved neural networks are then ranked by applying ElasticSearch. The design of system services and the N2Query are illustrated in the above figure.

[19] https://nlp.stanford.edu/software/

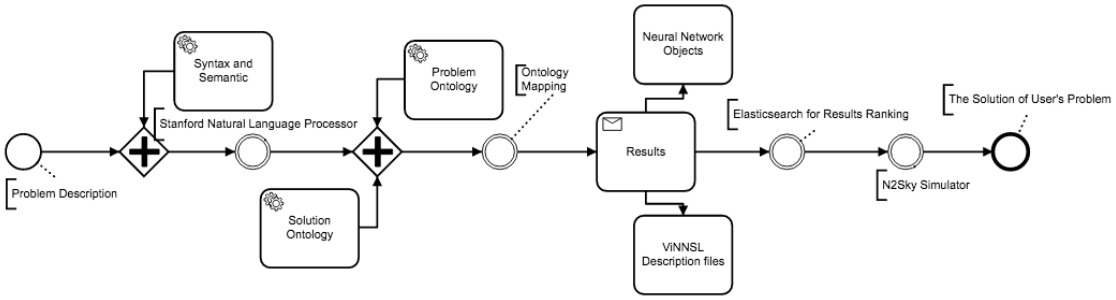


Figure 2.28: N2Query components control flow

The N2Query (Mobile) Web Portal The N2Query system gives users access to scientific knowledge through browser-friendly interfaces. N2Querry provides two different interfaces to the user, a free text and a directory search. Figure 2.31 shows, the N2Query free-text interface. The directory interface depicted in figure 2.33 allows a category search and offers a brief description of the mechanism of N2Query interface by providing some examples to show how the user can interact with N2Query interface.

Natural Language Processor Service This service analyzes the semantic definition behind the user query. The NLP service applies five steps (Lexical Analysis, Syntactic Analysis (Parsing), Semantic Analysis, Discourse Integration, Pragmatic Analysis) to understand the meaning of statements.

Problem Ontology service: The service is focused on a hierarchical system of previously existing neural networks. This service aims to identify the user's neural network problems into a possible classification of neural network problems: optimization, approximation, search etc.

Solution Ontology service: This aims to store all solutions offered by neural networks in a hierarchical arrangement in the N2Sky virtual organization. The ontology describes possible solutions for a particular problem.

NN Resource Architecture web service: This administers objects of neural networks, which are considered solutions to a particular problem, and could be utilized under the framework of the N2Sky simulator.

ViNNSL Description web service: This service is responsible for attributing N2Sky with ViNNSL description files. These files describe the structure of managed neural network objects used by N2Sky to create and train neural networks.

WordNet web service: WordNet is a big lexical database of the English language. Nouns, verbs, adverbs and adjectives are grouped into sets of synonyms (synsets), each expressing a distinct concept [Pri20]. This service delivers a list of words and synonyms to the NLP service to create lists of all user problems' synonyms.

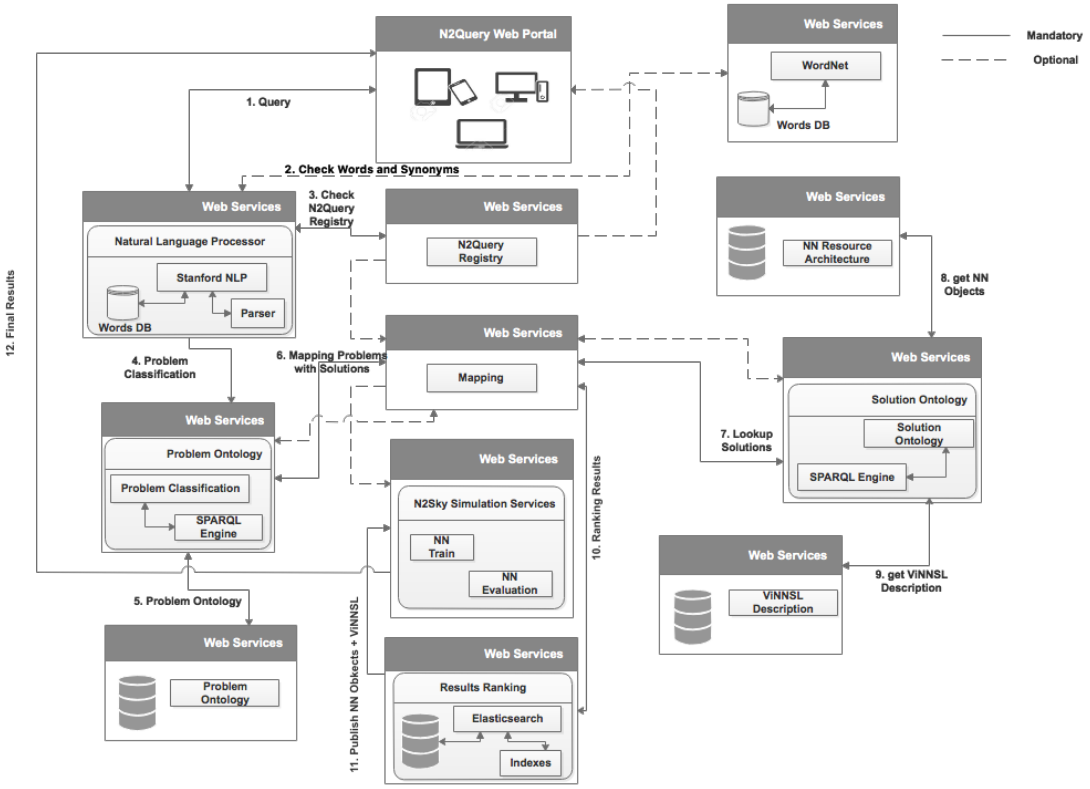


Figure 2.29: N2Query architecture and components

N2Query Registry: This service contains all well-answered queries. N2Query checks first if the user question is already asked before. If so, this service delivers the stored solution (NNs + ViNNSL) to N2Sky to retrain the network and get solution(s).

Elasticsearch service: This service is responsible for ranking solutions and publishes a list of filtered solutions.

Mapping service: This service provides the mapping technique to match a specific problem to possible solutions using a SPARQL query language and then deliver a list of solutions to that problem.

There are three ontology combinations: ontology linking, ontology mapping, and ontology importing [HS10]. For our problem, we apply ontology linking, where individuals from distinct ontologies are connected with links.

The concept is as follows: We administer basically two ontologies, a problem ontology and a solution ontology [HMS14]:

- The problem ontology is a hierarchical framework made of typical pattern of different application problems (e.g., optimization, classification, approximation, pattern restoration, cluster analysis, feature extraction, and others). In the ontology hierarchy, these main domains are finer distinguished until the single problem specifications show up in the leave nodes.

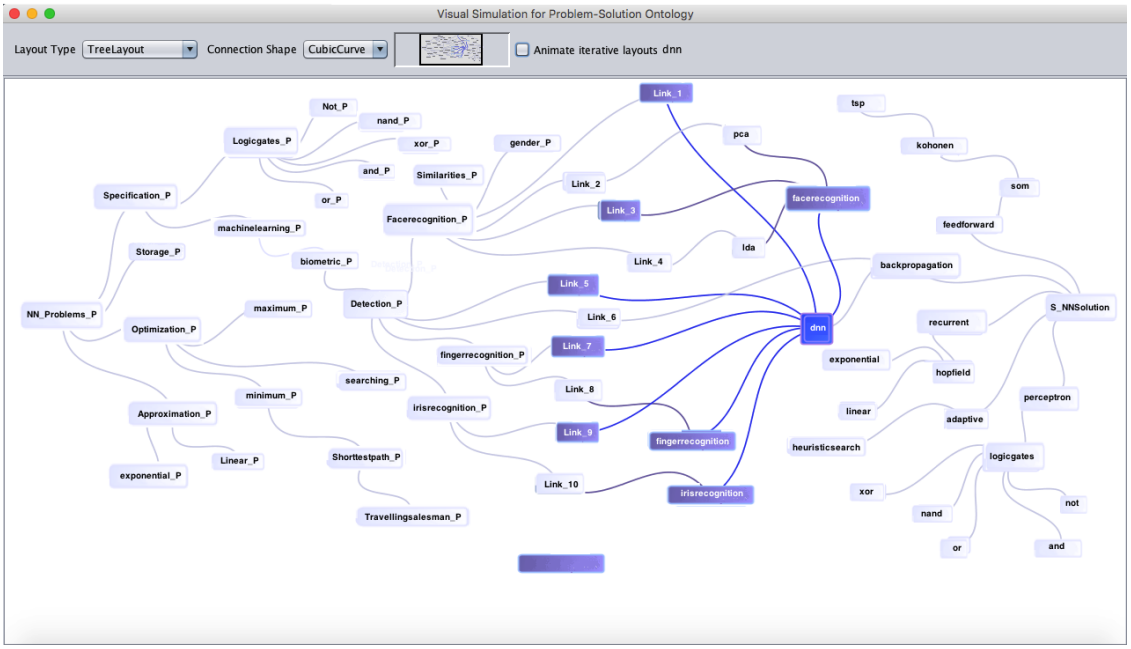


Figure 2.30: Mapping between solution ontology (right) and the problem ontology (left) for the "Face Recognition" Problem

- The solution ontology stores all known N2Sky neural networks organized according to their paradigm, as perceptron, multi-layer backpropagation, self-organizing maps (Kohonen cards), recurrent networks (Elman, Jordan, etc.), cellular neural networks, etc. In this way, the ontology provides a fine-grained structure, eventually offering neural networks as leaves.

Figure 2.30 depicts the two ontologies for solving the problem of "Face Recognition" (see case study section). Our N2Query component has already connected them (the nodes are connected with links). The links between problem-solution ontology simulate the mapping process between the defined problem with solution(s). We generate a mapping of problem ontology nodes, describing a specific problem, creating solution ontology nodes, and denoting network objects, which deliver a solution for this problem. Links can be defined not only between leaves of the hierarchies but also between internal nodes.

According to the information of the ViNNSL, the N2Sky maps out the problems to solutions automatically after the insertion of new network objects.

Integrating with the Ontologies This workflow for integrating new neural network resources into the knowledge repository of N2Sky can be described by the following algorithm A1 [SMM16].

Algorithm A1:

1. N2Sky Resource Provider (RP) attributes its Neural Network Resource (NNR) with a ViNNSL Description (VD) specifying structural and semantic information.

2. RP sends VD together with NNR or URI of NNR to N2SKy knowledge repository.

3. VD is integrated into Problem Ontology (PO) according to problem domain.

4. NNR or its URI is integrated into Solution Ontology (SO) according to network paradigm.
5. Link between VD insertion node in PO and NNR insertion node in SO is created.

Querying the Ontologies The search algorithm is as follows: Based on the user query's natural language keywords, a scan over the problem ontology is performed. Hits, patterns matching nodes in the hierarchy to the solution ontology are followed. A scan of the network objects, representing solutions to the problems, is done, and relevant results are reported to the user. A fitting rank of a problem can guide the sequence of the results to solution matches.In this way, the management service for the distribution of neural network sources is centralized. According to this approach, the number of network resources to be checked is cut dramatically by only checking resources targeting the problem domain [HMS14]. The generic workflow for executing a query on the knowledge repository can be described by the following algorithm A2 [SMM16].

Algorithm A2:

1. User describes in natural language his/her Problem Description *PD* using N2Query interface.
2. Cognitive representation of the problem description delivering a synset *SS* (set of cognitive synonyms).
3. Problem description is classified according to *SS* in *PO* delivering set of *PO* nodes.
4. Links from *PO* nodes to respective *SO* nodes are followed.
5. *SO* nodes are starting points of tree search delivering URIs of possible solution candidates.
6. *VD* of solution candidates are analyzed and ranked according to match with *PD*.
7. Ranked list is reported to user.

N2Query Case Study

In this work, we choose the problem of Face-Recognition as a case study example [HSYC13]. In our use case, the user submits a query "How to solve face recognition problem" as shown in figure 2.31. The query is then processed and the results are displayed in the same interface.

Our approach to solve this problem consists basically of two phase, the ontology integration phase and the ontology query phase.

Ontology Integration Phase

The prerequisite for the user query is that the required neural network object must be present in the ontology architecture as well as the required ontology mapping should already be available within the system. The search process is performed on this ontology architecture. In the integration phase a neural network resource, e.g. a trained neural network object, which is provided by a member of the N2Sky virtual organization, is entered into the solution ontology. We use as a running example the face recognition problem [HSYC13]. It is assumed the respective backpropagation network was realized and contributed to N2Sky. Hereby the algorithm A1, presented in section 2.3.5, is executed:

The provider of the NN resource, the Backpropagation network, uses the ViNNSL language for the description of the problem and paradigm domain in step A1/1. Hereby, the `paradigm` and `problem domain` tags of ViNNSL are used.

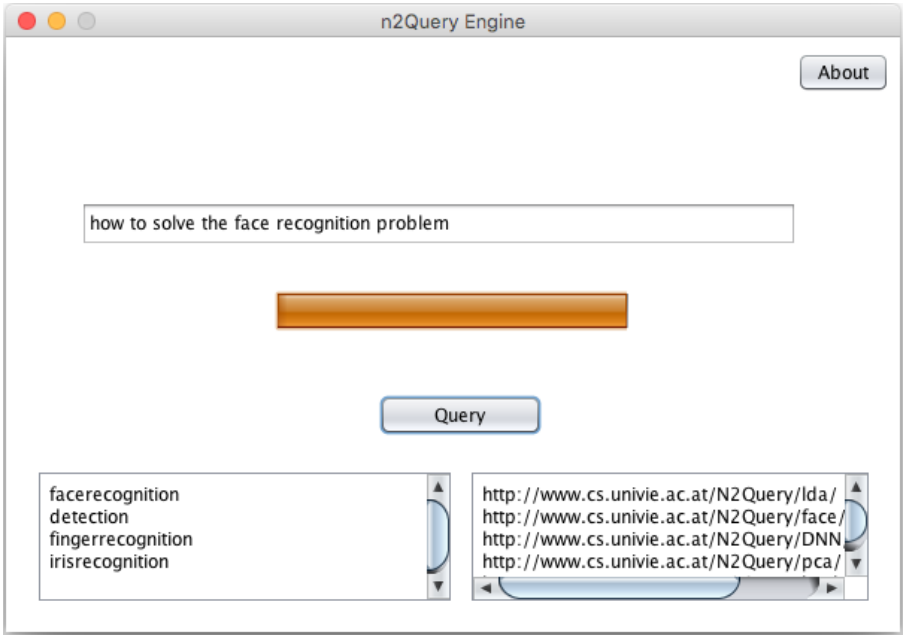


Figure 2.31: N2Query free text user interface

In step A1/3 the description of the NN resource, Classification → MachineLearning → Biometric → FaceDetection → Gender, is integrated into the problem Ontology. The NN resource is integrated into the Solution Ontology accordingly to its paradigm family in step A1/4, Backpropagation → DNN → FaceRecognition → PCA. In step A1/5 an appropriate link from problem to Solution Ontology is created pointing from problem description to the respective physical NN resource, see figure 2.30.

Ontology Query Phase

In the following we show how a natural language free text user query is analyzed by the N2Query system.

The following query analysis steps refer to the workflow from figure 2.29, marked as ordered numbers integrating various components of the N2Query architecture.

1. The user query is sent to the Natural Language Processing (Stanford Parser) web service to analyze the semantic of that query.

2. The NLP web service connects with WordNet web service which delivers a list of words and synonyms of the user problem.

3. If the query has been asked before, the N2Query Registry service sends all details to the Mapping service. This service is responsible for gathering neural network paradigms as solutions of that problem. Afterwards, the Mapping web service publishes solution(s) to N2Sky for retraining networks.

4. The Problem Ontology service receives the recognised statement of the user query in form of tokens. That service classifies the user problem under the hierarchal structure of the most known neural network problems.

5. The SPARQL query algorithm is applied on the problem ontology to match the user problem with stored problems. Conclusively the SPARQL query engine sends all matched classifications to the Mapping service.

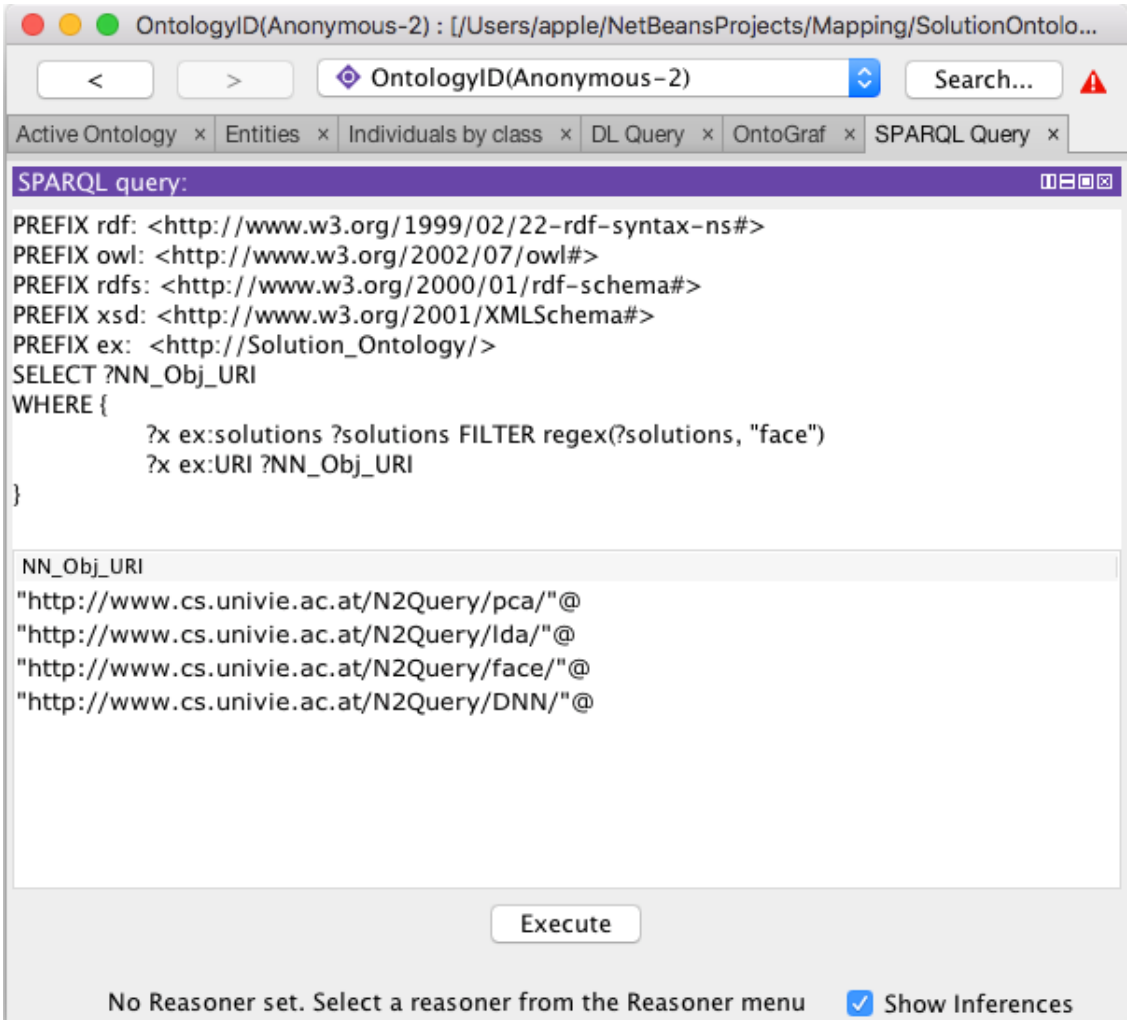


Figure 2.32: SPARQL query and list of URI's of NN objects as solutions using protégé

6. The Mapping service matches the problem(s) to the solution ontology. Figure 2.30 shows the mapping between solution ontology and the problem ontology for the face recognition problem.

7. Solution Ontology service looks up the solutions by a respective SPARQL query.

8. The Solution Ontology service gets the respective neural network objects from the NN Resources Architecture service. Figure 2.32 shows the SPARQL query and the list of URIs of neural network objects as solutions using Protégé ontology tool.

9. The solution Ontology service receives also the respective ViNNSL description file(s) which describe the received neural network objects.

10. The Elasticsearch service is applied on the received solution(s) for ranking and filtering results and publishes a list of solutions of the user problem to the N2Sky simulator.

11. N2Sky receives the published solution(s) (NN objects and ViNNSL) and starts creating, training, retraining and evaluating neural networks for the user problem.

12. N2Sky sends the final result(s) to the user as shown in the bottom right pane in figure 2.31. Figure 2.33 represents another possibility to present results in a structured directory interface.

In this process, we use the Stanford parser for Natural Language Processing. The problem-solution ontologies are implemented by RDF and processed by the OWL semantic web languages [SS13]. Huge storage repositories for RDF data have been developed, which store the RDF triples in a relational database (RDB) [BHS03]. So, we use the Eclipse RDF4J Framework for ontology storage. According to the steps in the last section, N2Query starts to map from the solution to the problem ontology to retrieve the existing solutions to that problem. In this process we use SPARQL language for the matching process. Thus, specifically the algorithm A2, see section 2.3.5, is executed by applying sematic web techniques:

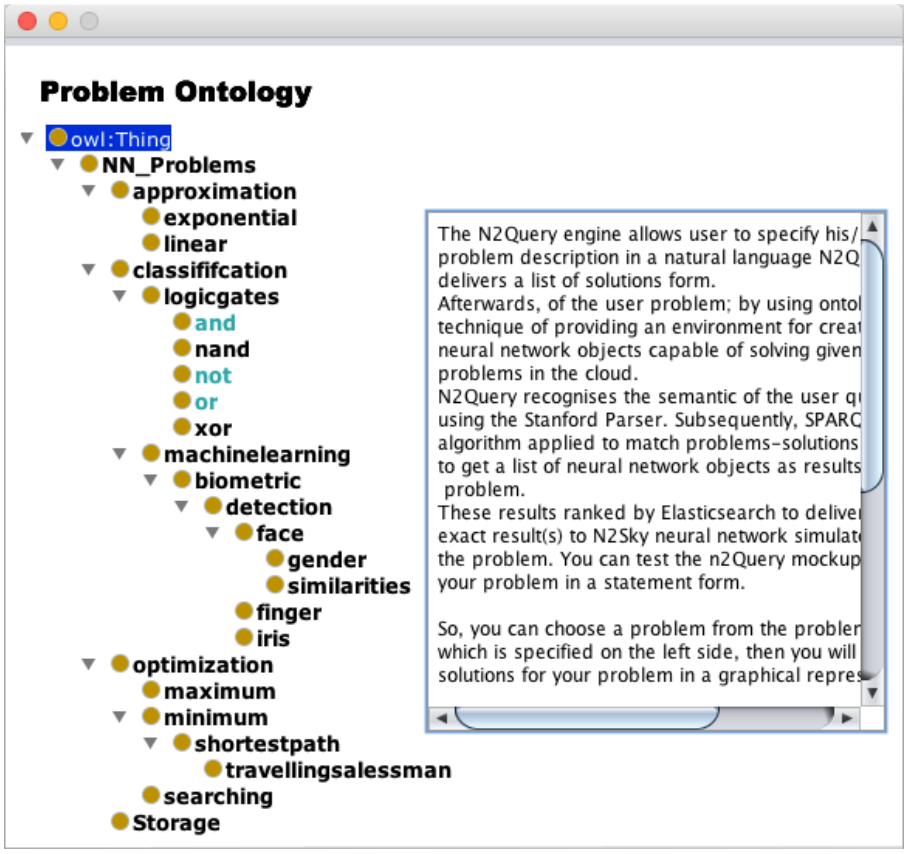


Figure 2.33: N2Query directory and guide interface

In step A2/1, querying the N2Query tool by a natural language phrase like, "How to Solve the Face Recognition Problem", the N2Query system tries to recognise the semantic representation of this problem using Stanford natural language processor (step A2/2), and receives in the synnet set a phrase like "Face Recognition". In step A2/3 the SPARQL Query algorithm classifies the

user query into the problem domain, and, following the link to the Solution Ontology (step A2/4), delivers by a subtree traversal (A2/5) the possible URIs of the existing NN objects. Based on the ViNNSL descriptions in step A2/6 the system produces and reports a ranked list of qualified solution URIs to the user's problem (A2/7).

2.4 Ontology Approach in Cybersecurity

The ontology approach has been proposed in several works in the cybersecurity domain [MSG+19]. A systematic review of the existing ontological engineering researches applied to the security is discussed in [BLV+08]. Ref. [MJGA15] proposed a reference ontology to help in finding security solutions to the Internet of Things (IoT) environment. The proposed reference ontology is based on the modeling process to unify concepts and explain relationships among the main components of risk analysis of information security. Ref. [MAG+18] introduced an ontology-based framework to control business process for IoT security.

Ref [Wil18] develops a collaborative system based on the ontology approach. This collaborative system helps in security requirement elicitation for software engineers. A method for security requirements engineering of security and domain ontologies is presented in [SSWM13]. This method is based on a collection of rues are applied to the ontologies to extract relevant security and domain knowledge. Also, Ref [SSMCW15] presented a general security ontology for requirements engineering. The authors implemented and developed an interactive environment for the ontology. This environment assists in using the ontology through the security requirements engineering process.

2.5 Chapter Summary

This chapter presented relevant research approaches in automotive cybersecurity. The chapter started with explaining the different levels of vehicular automation and then introduced a brief discussion on the critical safety engineering process in the automotive sector. Afterwards, the cybersecurity topic was included within the chapter context to clarify the importance of cybersecurity in the automotive domain and how it is essential in the vehicular development life cycle to protect the critical assets, including managing the functional safety in the vehicles from different cyberattacks. The section on cybersecurity discussed the common research works and projects that have been conducted on automotive cybersecurity. The chapter also discussed some other relevant topics such as threat modelling, the security requirements engineering process, and security standards. The next chapter discusses the research methodology approaches to describe the structure of the proposed cybersecurity framework in the automotive domain.

3 Research Questions and Methodology Approaches

This chapter introduces the research design methods pursued in this research context to come across the research finding to address the main research problem. As discussed in [HRM+04], the design-science paradigm aims to increase the organizational and human capabilities by defining, developing, and creating new innovative artefacts. In order to build a problem-solving approach for addressing a particular research issue, it is essential to clarify the main research gaps and define the relevant facts and hypothesis. The research facts describe a set of observations according to a particular research issue; where as the research hypothesis describes the primary concept of the research work based on a set of proofs and tests. Furthermore, this chapter identifies the research design and methodology utilized in this study to create innovative artefacts based on a combination of research actions (i.e., develop solutions for a particular problem based on its diagnosis [Bus20]) and collect research data that assists in achieving the primary research objective. Also, it discusses the research analysis according to the ontology hierarchy for solving the main research problem as it is focused in this research context. The chapter then describes the relationships between various entities in a vehicle (e.g., components, assets, security measures, threats, vulnerabilities, security requirements, and relevant risks), which aims to clarify the proposed cybersecurity framework's reliability and feasibility to address the main research issue. Figure 3.1 depicts the main structure of the methodology framework that followed to design the proposed cybersecurity framework in the automotive domain.

This chapter includes content from some articles published within this thesis work. The following list refers to the selected publications that are integrated within the context of the chapter:

- *Abdelkader Magdy Shaaban, and Schmittner Christoph. 2020. "ThreatGet: New Approach Towards Automotive Security-By-Design." Pp. 413–19 in IDIMT-2020 Digitalized Economy, Society and Information Management. Kutná Hora, Czech Republic.*
- *Abdelkader Magdy Shaaban , Christoph Schmittner, Thomas Gruber, A. Baith Mohamed, Gerald Quirchmayr, and Erich Schikuta. 2019. "Ontology-Based Model for Automotive Security Verification and Validation." Pp. 73–82 in Proceedings of the 21st International Conference on Information Integration and Web-Based Applications &; Services, iiWAS2019. New York, NY, USA: Association for Computing Machinery.*
- *Abdelkader Magdy Shaaban, Christoph Schmittner, Thomas Gruber, A. Baith Mohamed, Gerald Quirchmayr, and Erich Schikuta. Ontology-Based Security Requirements Framework for Current and Future Vehicles. Vol. Data Science and Big Data Analytics in Smart Environments. CRC Press Taylor & Francis Group. - (Accepted)*

Figure 3.1 illustrates the primary research methodology approaches inspired by [HRM+04], to create innovative artefacts and alternative research processes for achieving the main research goal

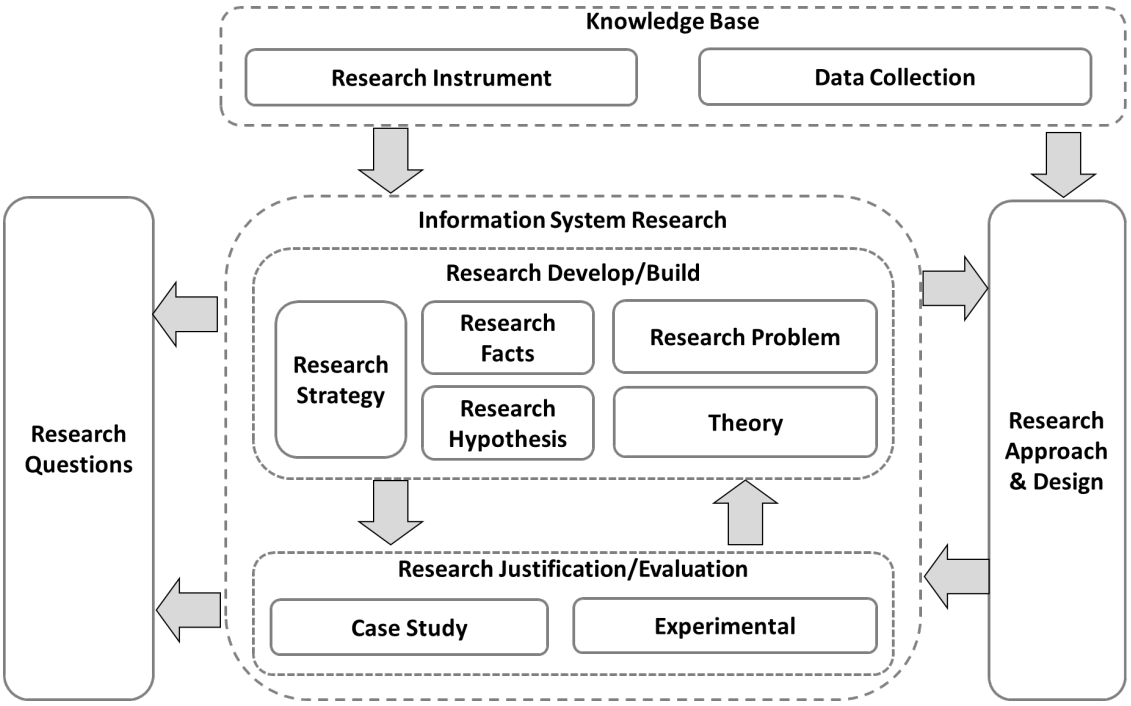


Figure 3.1: The main structure of the research methodology framework

and answer the open research questions. According to the [HRM+04], to design science as an Information System (IS) research, we need to design both a set of activities and associated research artefacts. Both are used in representing the main research objectives and define a straightforward design to establish a problem-solving paradigm that enables switching between design processes and research artefacts. Throughout this process, the design-science research shall evolve the design artefacts as a part of the research context. Through this research activity, various processes and artefacts are developed and designed to achieve the primary research purpose.

3.1 Research Questions

The overall research objective of this thesis is to design an automotive cybersecurity framework to support in verifying and validating the correctness and completeness of the applied security requirements within the vehicular units. The framework also aims to compensate for uncoverd security gaps with relevant security requirements, which protect vehicle components/assets from various cyber-attacks. Therefore, we can formalize the main contribution of research points into a set of research questions to improve the comprehensive knowledge on this research topic:

- ***RQ1. How can an ontology-based cybersecurity framework for the automotive domain be built?***
  - ***RQ1.1. How can security vulnerabilities in the vehicular domain be treated focusing on the interaction between the internal and external units in the vehicular systems?***

  In order to cope with the security issues in the vehicular system, we need to understand potential threats and relevant security requirements that apply to protect vehicular compon-

ents/assets. The common feature between security problems (threats) and security solutions (requirements) is the applied security measures for each vehicular internal/external component/asset. These measures are defined as a set of protection mechanisms for each vehicular element against multiple cyber attacks. However, these measures could be represented as a set of security vulnerabilities when attackers exploit them to reach a particular malicious goal.

A well-established ontology structure helps find the common security characteristics between vehicular components/assets (internal/external), threats, and security requirements. Section 3.6 gives a deep insight into the structure of the proposed ontology model to define a clear understanding of relationships among instances represent a vehicular ontology model (i.e., components, assets, threats, security requirements, and security measures). Then the framework applies a set of inference rules to deduce the most applicable security requirements according to the existing security vulnerabilities to address the identified potential threats and protect the vehicular units from different cyberattacks. Section 6.4 explains the main activities of the proposed validation process that integrated into this research work. However, the primary concept of the proposed validation process is based on the risk management process by ISO 27005, according to ISO 31000. This risk management process handles security vulnerabilities by selecting the most appropriate security requirements to solve the existing security issues. This process identifies all potential risks that could affect the vehicular systems. Then defines the exact potential threats for investigating the exiting security vulnerabilities in vehicular systems according to the applied security properties. The risk process then evaluates the identified risks, to estimate the severity of risks, which helps select an appropriate set of security requirements that can address these risks.

Consequently, this process is based on the complete established relationships between the vehicle entities and existing security vulnerabilities, that could be handled by selecting the applicable security requirements. The framework handles that by using a rule-based approach based on a set of inference rules, to improve the quality of the outcomes of the process and deduce most suitable security solutions for the actual security problems in the vehicular design.

- ***RQ1.2. How can an appropriate ontology model support in the automotive cybersecurity be built?***

Cybersecurity in the automotive domain becomes one of the essential parts of the vehicle development lifecycle. It is applied to protect personal data such as contacts, credit card information, personal accounts, and other data collected by the car. It is also responsible for protecting vehicular elements that are managing and monitoring functional safety in vehicles. Therefore to build and design a secure vehicle infrastructure, we need to:

- identify the exact potential threats;
- select an applicable set of security requirements accurately.

Accordingly, Chapter 4 and Chapter 5 explain a complete overview of the research data collection that is specified, adapted, and incorporated in this thesis work to create a comprehensive ontology hierarchy in automotive systems. The ontology model consists of a wide range of the most common potential threats in the vehicle domain, that affect various vehicle components/assets. It also includes a broad collection of security requirements

from various resources to protect vehicle components/assets from various cyber-attacks and address existing security risks.

- ***RQ2. What are the main activities that should be integrated within this work to implement a reliable cybersecurity framework in the vehicular domain?***

  One single vulnerable point in a vehicle could make all components or the whole vehicle exposed to a higher degree of cyberattacks. Furthermore, it is essential to understand the actual security weaknesses in a vehicle at the early stages of the security engineering process because once the vehicle is built, it becomes more difficult to add security. Regarding the complexity of vehicle architecture, it is essential to develop new advanced approaches that are fully-matched and fully-adaptable with different input forms representing the vehicular architecture design.

  The proposed ontology-based cybersecurity framework consists of a set of activities described according to a set of standardized criteria to ensure the quality of design of the proposed framework. As addressed in RQ1, the ontology approach is integrated into the work to create a complete taxonomy model of the vehicular structure with all relevant security details (i.e., security measures, security requirements, potential threats, security levels, and severity levels of risks). Ontology is considered a robust approach that uses standard specifications for knowledge representation such as vocabularies, taxonomies, classes, individuals, and annotations. The ontology hierarchy is built according to multiple reliable resources to guarantee the quality of the research context. The representation ontology model of potential threats is defined from the UNECE threats list, V2X HSM protection profile, and common vulnerabilities and exposure - CVE. As with the security requirements, a set of data sources are used to define the applicable security requirements in this research context, such as the IEC 62443-4-2 security standard, and the security requirements from the V2X HSM protection profile based on the common criteria. Section 3.4.1 gives an overview of that; with details specified in Chapter 4 and Chapter 5.

  The proposed framework consists of a set of activities designed to make the proposed framework fully adaptable for multiple ontology input forms. It has five main activities, including data digestion, reading, scanning, and understanding the consistency between the relationships of the vehicular ontology model. Then the verification activity is developed according to the logical verification based on the theorem proving method. The verification uses this method to logically check the compatibility of the model design and detect existing gaps. The logical verification's primary goal is to ensure the logical correctness of the security characteristics within the ontology structure. The validation activity is designed and identified according to the testbed execution method, according to the risk management process defined by ISO 27005 according to ISO 31000. The validation approach follows the main four steps of the risk management process (i.e., risk identification, risk analysis, risk evaluation, and risk treatment) to estimate the degree of correctness and completeness of the applied security requirements vehicular components/assets. This approach defines the exact security weaknesses in the vehicular design. It defines the applicable set of security requirements to address these security issues and protect vehicular units from different cyber malicious activities. The security gap analysis and enhancement activities are assigned as the last two main activities applied by the proposed framework. The actual goal of these activities is to address the previously discovered security issues by the verification and the validation activities. Therefore, the cybersecurity framework verifies and validates the

applied security requirements and handles security issues by filling the existing security gaps and improving the overall security achieved to ensure the main security target. Chapter 6 introduces the metamodel of the proposed cybersecurity framework, which is considered the primary core of this dissertation work.

- ***RQ3. What is the main research impact on the automotive domain?***
  - ***RQ3.1 What are the main impacts of the proposed framework on the cybersecurity engineering process in the automotive domain?***

According to the high impact of cybersecurity in the automotive sector, cybersecurity plays an integral role in the automotive industry. Cybersecurity is the practice of protecting components and software that manages the functional safety in a vehicle from different attack scenarios (i.e., unauthorized access, information disclosure, man-in-the-middle, or others). In the future, the manufacturers of vehicles will need to check the cybersecurity of their vehicle systems before their products can be accepted for sale.

The framework presents an advanced concept of managing security issues in the vehicular domain by providing complete automation actionable to save time and effort spent by the automotive security architect. The security architect typically spends much time and effort to match the common security properties between threat, assets, components, and relevant security requirements to protect vehicles from different cyberattack scenarios. This process is achieved successfully by the proposed ontology-based framework, within a particular time with accurate findings. In comparison, the manual approach could not reach the same accuracy as the proposed framework, and unreasonable efforts would be consumed.

The proposed framework is designed, developed, and implemented to be fully-adaptable for various kinds of ontology model inputs representing the interactions between different components/assets in the vehicle network. It suggests the most applicable security requirements to address security issues and protect the vehicle components/assets. Security requirements are selected according to security mechanisms and are then are applied within the vehicle design. The framework can handle different types of input forms of the vehicular ontology representation model, such as follows:

- Integration of both threats and security requirements.
- Import of potential threats.
- Import of security requirements.
- No remaining threats or uncovered security requirements.

Chapter 7 discusses more details on the security activities of the proposed framework in the automotive domain.

- ***RQ3.2 How can the effectiveness of the cybersecurity framework in the automotive domain be demonstrated?***

As discussed in RQ.3.1, the framework can handle multiple data inputs of the vehicular ontology representation model, by selecting a set of suggested security requirements applicable for addressing the existing security issues. The framework also verifies and validates the correctness of the applied security requirement(s) into a particular vehicular asset/component. These points are demonstrated in Section 7.3.1, the framework checks the accuracy of the selected security requirements for a particular vehicular asset, and then

checks if relevant security requirements handle all the imported threats. In the experimental outcomes, we integrate security details of the selected asset (i.e., VCS data) within the input of the ontology representation model as described in the V2X HSM protection profile. The outcomes of the cybersecurity framework are matched with the same data in the protection profile document.

As mentioned before, the framework is designed to be fully-adapted with multiple data input forms. It accepts the outcomes (i.e., identified threats or security requirements) from other external tools or manually selected data. The framework's security KB is considered an essential part of this process, which contains all the security requirements that address security issues. In importing the threats from external approaches, the framework studies these threats and deduces the most applicable security requirements fit these security issues. Then the framework gives an initial estimation value that reflects the degree of effectiveness of the selected security requirements against the existing potential threats. In order to prove the effectiveness of the framework in selecting the security requirements, we keep a small set of security requirements in the KB, and apply the framework once again to the input ontology model. The framework chooses some new security requirements to fit in solving existing security problems based on the available security requirements in the KB. This experimental scenario is discussed in detail in 7.3.2.

Similarly, in the case of importing the security requirements from external resources as discussed in Section 7.3.3, the cybersecurity framework checks first if these requirements address the real exiting security issues or not. This process follows reverse engineering activity by deducing potential threats (security issues) from the existing security requirements (security solution). Afterwards, the framework investigates the impact of the previously selected security requirements against the newly inferred potential threats. According to outcomes, the framework performs the decision on the correctness of these requirements. This example demonstrates that the framework also could be applied to define strategic automation for generating test sets.

The framework also can handle the vehicular input model, with the absence of both threats and security requirements. It follows the main risk management activities to identify, analyse, evaluate, and treat the existing risks. According to the outcomes of these activities, a set of potential threats are discovered, and relevant security requirements are generated to address these issues and protect the target vehicular asset/component. This point is discussed and presented in Section 7.3.4.

We claim that this research work introduces a fully-adaptable cybersecurity framework for the automotive domain regarding all the discussed cases. The framework helps verify and validate the correctness of the applied security required within the vehicle system design. It also focuses on the most proper security requirements that contain the common security measures with threats and particular assets/components in the vehicular network. A summary of the overall work evaluation is discussed in Section 7.4.

3.2 Research Develop/Build

As presented in [HRM+04], IS research provides fundamental theories, framework, instruments, methods and models used in the develop/build phase of the research study. In this work, we

define a set of phases and sub-phases of research activities to clarify the primary research strategy approach to create new innovative artefacts for solving the main research problem.

3.2.1 Research Problem

Regarding the specification of the UNECE WP29 for the connected and autonomous vehicles, the automotive manufacturers should design their cars to be more cyber-secure. In the future, they should certify their automobile's compliance to confirm that their vehicles meet a particular security level and are permitted to be placed on the market. Also, the absence of the security requirements in this domain leads to integrating other security standards from other relevant domains. However, there is still a question: how do we implement and integrate these requirements within the vehicle assets/components level? The other point is, modern vehicles are no longer considered as mechanical parts connected over gears. However, modern cars contain several electronic/electrical units interacting and communicating together through the vehicular network via different protocols. The diversity in communication protocols and the heterogeneity of electronic parts in the vehicle lead to a rise in the proliferation of security vulnerabilities, which leads to unwanted consequences. The traditional security verification and validation methods become more complex and not applicable for verifying and validating the security requirements in modern vehicles. Instead, verification and validation approaches need to be aware of all vehicle components/assets, potential threats, and associated security requirements, which is considered a challenging process.

The proposed methodological approach aims to study the relationships between various entities in the vehicular design to understand the relations of vehicular components/assets with potential threats and required security requirements. This approach helps in improving the verification and validation approach. In addition, it assists in selecting a suitable set of security requirements precisely to protect vehicular components/assets against potential threats. Therefore, the need for design such a methodological approach is required to address the discussed research problems.

3.2.2 Research Facts, Hypotheses, and Theory

Modern vehicles have a broad range of electronic and electrical systems to handle and control vehicle functions. Integrating internet technology with modern cars exposes these vehicle types to cyberattacks. Similar to any connected device to the Internet, modern vehicles are exposed to be attacked by different malicious activities. Malicious activities are defined as events performed by an attacker to perform and reach a particular malicious goal. In addition to that, inside a modern vehicle is a hidden complexity of up to 150 Control Units connected via multiple communication networks, controlling many sensors and actors with an increase in automated decision making. Installing malicious software through the vehicular ports could have unwanted consequences. For example, "Install malware on the infotainment unit" as mentioned by Craig Smith in "The Car Hacker's Handbook - A Guide for the Penetration Tester" [Smi16]. The security requirements are highly recommended as a set of non-functional requirements that need to be adequately identified to protect the vehicle from different malicious activities. Therefore, cybersecurity requires to be a part of the developing phases in vehicular manufacturing. All these points are considered a collection of facts used to construct the key research context based on realistic automotive observations in cybersecurity.

According to this research context, we can define the research hypothesis as a lack of security protections in vehicle design, leading to different negative consequences. For example, as dis-

cussed above, installing malicious code into the vehicle could negatively impact the vehicular network. Therefore, implementing the input validation and input sanitization, for example, in a vehicle vulnerable point could be an adequate protection mechanism to reduce the possibility of this malicious action. Thus, we can design a secure vehicle when we clarify existing security vulnerabilities that could be exploited by different potential threats and accurately select relevant security requirements to address these security issues. These research hypotheses will be discussed later as a set of potential threats that threaten the security protection mechanisms of the vehicular network. A real case study is integrated into this work to verify the reliability and feasibility of this research. The proposed cybersecurity framework will be tested in various experimental outcomes to prove its effectiveness in automotive cybersecurity. In the case study, four different examples are introduced to illustrate the proposed framework's flexibility to be incorporated and identify the protection criteria for the existing security vulnerabilities in vehicular design; this will be covered in Chapter 7.

Now it is essential to combine these facts and hypotheses to define the main theory of this research. Therefore, clarifying the relationships between vehicular units is essential to comprehend the main security issues in vehicular design and highlight the existing security weaknesses. This helps the research to create advanced techniques for improving the accuracy of the verification and validation processes. It also supports to increase the efficiency of selecting applicable sets of security requirements for protecting the vehicular assets/components and addressing the potential existing threats.

3.2.3 Research Strategy

The design-science paradigm is considered the other way of the information system research cycle to develop and assess the information technology artefacts for solving particular problems, as discussed in [HRM+04]. The design-science paradigm is followed in this thesis to create new innovative artefacts that assist the research process in reaching the primary goal of this study. The qualitative research intends to collect and analyze text, video, or audio (i.e., no-numerical content) data to understand the primary conception of the research method, as discussed in Scribbr [1] [noa20b]. Therefore, the qualitative data method is integrated into this research process to collect and analyze non-numerical data used to get insights into the cybersecurity issues in vehicles and generate innovative artefacts to meet this research's primary target.

3.3 Research Justification/Evaluation

The research justification and evaluation activities are used to assess the main research activities and identify weaknesses in the theory or the research artefacts, which need to be improved or refined. In this work, four different experiments were created on the selected case study to evaluate the robustness and the correctness of the proposed framework. Throughout the justification/evaluation phase, there are multiple weaknesses discovered and enhanced until we overcome the most identified weaknesses in our research activity. For, example, the outcomes of the validation process as discussed in Section 6.4 were not satisfied in some cases, especially when the integrated security requirements within the ontology input were not completely matched with the inferred list of requirements. After some experiments on the applied case study, we enhanced the validation process capability to check the correctness of the previously selected security requirements, which

[1] https://www.scribbr.com

could be defined by any security requirements management approach. This action created a new research artefact for defining a new reverse action from solution (security requirements) to problems (threats) to examine the impact of these security requirements against the inferred list of threats. This scenario is discussed in detail in Chapter 7.

3.4 Knowledge Based

The knowledge base contains the key tools from and through this research work. The knowledge base includes a collection of fundamentals and methodologies as discussed in [HRM+04]. In this work, the knowledge base consisted of data gathered to accomplish this research activity. It also has a set of instruments used by the develop/build phase to establish different research processes and create new research artefacts.

3.4.1 Data Collection

The qualitative data analysis supports the research findings based on interviews, observations, and report records, as mentioned in [JS19]. The qualitative research method analyzes the non-numerical data collected and integrated into this work to create innovative artefacts. In this research, two distinct data sets from various sources are collected and analyzed for this research cycle; the collected data are classified into potential threats (i.e., problems) and security requirements (i.e., solutions) for addressing these problems. As discussed in [noa20b], these data are considered secondary research that aims to collect already existing data. The following Table 3.1 represents the data collected and used within this research cycle.

Chapters 4 and 5 discuss in more detail about analyzing these data to extract common security properties for building the ontology hierarchy structure. Common security properties are defined and deduced from these data to define the relationships between the components, assets, threats, vulnerabilities, and security requirements in vehicles. Relationships are developed to build an ontology-based taxonomy for creating the primary context of the research work.

3.4.2 Research Instrument

The research instrument section describes the tools used in this research to analyze, develop, or construct the different phases of this research process.

Ontology: In this work, the ontology approach is proposed to pave the way for checking the security compliance in the vehicular domain. The ontology is considered a robust methodology for managing a wide range of artefacts in the automotive sector to see whether sufficient security mitigation is applied correctly in the early stages of the system design or not. The concept of the ontology is a topic correlated to the study of philosophy; it is used to describe the world we live in, since the time of Aristotle. The ontology concept is used in computer science to describe a new methodology of better understanding the digital world. The term ontology in the computer world is similar to the philosophy, which represents a set of primitives to model a particular domain of knowledge. The primitives are typically classes, attributes, and relationships. The representational definitions of these primitives contain information about the meaning and logical consistency [Gra08]. The ontology provides various interlinks among ontology nodes to define relationships between different entities [HS10].

Table 3.1: The types of collected data in this research process

Data Type	Document Type	Document Title	Notes
Threats	Potential Threats	UNECE, United Nations Economic Commission for Europe, CSⅣOTA ad hoc "Threats 2" CVE - Common Vulnerabilities and Exposures	A set of high-level of potential threats and relevant vulnerabilities in the automotive domain, from the required UNECE vehicle [UNE17b] A library for known security vulnerabilities [Com20]
Security Requirements	Security Standard	IEC 62443-4-2: Security for industrial automation and control systems - Part 4-2: Technical security requirements for IACS components	A set of security requirements for the component level, as described in [IEC19]
Threats & Security Requirements	Protection Profile	Protection Profile V2X Hardware Security Module - CAR 2 CAR Communication Consortium	Define a set of potential threats and selected security requirements based on the Common Criteria [ISO09a], for the V2X HSM component as described in [Car19] [Car18]

RDF: Resource Description Framework (RDF), uses mainly the Extensible Markup Language (XML) format but also allows semantic [Thu15]. RDF presents a way to model knowledge but does not explain what it means [HFBPL09].

RDFS: RDF Schema (RDFS), includes a particular vocabulary for RDF to describe taxonomies of properties, classes, domains, and ranges for context requirements [HFBPL09].

OWL: Web Ontology Language (OWL) presents a vocabulary that expresses ontologies that encompass the domain knowledge semantically [HFBPL09].

SPARQL: An abbreviation of SPARQL Protocol and RDF Query Language. It is a query language for the RDF language. There are other sets of RDF query languages such as RDF Data Query Language (RDQL) and Sesame RDF Query Language (SeRQL); however, this research focuses on the use of SPARQL in executing queries on the RDF [HFBPL09].

SWRL: Semantic web rule language is often used to translate data from one ontology to another. The language specifies a combination of facts with other groups of facts. It also includes a library of multiple built-in functions to provide different operations on mathematical procedures, manipulations of strings, and logical operations of a set of variables in the ontology hierarchy [HFBPL09].

SQWRL: Semantic Query-Enhanced Web Rule Language is based on the SWRL rule language. This language uses SWRL rules to handle it as a pattern specification for queries [OD09].

SWRLAPI: The SWRLAPI [2] is used in this work for supporting the semantic reasoning process. The SWRLAPI is a Java API for supporting OWL-based SWRL rules and the SQWRL query statement and uses OWLAPI to manage the owl ontologies [O'C19]. The SWRLAI provides an authoring environment and programming interfaces to support building or rule-driven applications. The SWRLAPI has multiple components, including [OSM+08]:

- **Editor:** An extension of the Protégé-OWL for creating, editing, reading, and writing of SWRL rules.
- **Rule engine bridge:** It aims to incorporate rule engines and OWL reasoners into Protégé-OWL to execute SWRL rules.
- **Bridge:** Defines libraries of built-ins; it provides a mechanism for importing the built-in libraries and provides a mechanism to run the built-ins.
- **Libraries:** Set of built-in libraries.

Protégé: An open-source ontology editor for building smart systems [noa20c]. Protégé is used to build the structure of the ontology based on the previously discussed data sources.

Cellfire: A Protégé plug-in used to assist the importing of data from spreadsheets into OWL ontologies [joh18]. This tool will be discussed in Chapter 4 and Chapter 5 to define how the data is extracted from the different data sources and how we analyze these data to build up the hierarchy of the ontology structure.

3.5 Research Approach and Design

Cybersecurity plays an integral role in a vehicle environment because it aims to protect vehicle functional protection modules and technology against different attacks [NHT19]. Furthermore, it is essential to develop security requirements for vehicle components to ensure their reliability and security in the current and future vehicle industries to improve operational safety. The current standards for the safety and security of connected vehicles have several gaps. These standards are fragmented and inadequate [SSMG16]. Determining whether a system is entirely secure is very difficult. A system may be fully operational but can be vulnerable due to other unforeseen tasks during the process [KN17]. Modern vehicles are considered sophisticated systems that include numerous electrical units and communication protocols. Consequently, cybersecurity in the vehicular industry quite challenging. Any vulnerable unit in a vehicle may lead to complete attacks on the whole vehicle. New methodologies should be created that can manipulate an enormous amount of vehicle data to ensure a high level of protection in modern and future vehicles. These methodologies help to find the most relevant security requirements for the development of a secure vehicle. The complexity lies in manipulating data, where a large number of vehicular data have to be appropriately managed, processed, and manipulated.

[2] https://github.com/protegeproject/swrlapi

3.5.1 The Structure of the Ontology Model

This section introduces the structure of the ontology approach for managing all of the vehicular data and creating complete relationships between the vehicle's different entities (e.g., assets, threats, vulnerabilities, security properties, and security requirements). The well-designed ontology structure helps in facilitating the process of verifying and validating the chosen security requirements against the identified threats. Also, it improves the accuracy of selecting other security requirements able to address the potential threats and fill the existing gaps of the absence/exploitation of security measures that use as protection mechanisms for protecting vehicles' assets/components from alternatives cyberattacks. Furthermore, all vehicle data will be represented in classes, sub-classes, individuals, properties, and annotations of all assets, detected potential threats, detected vulnerabilities, and the selected security requirements with all related security properties. The ontology describes a set of primitive members in order to design a knowledge domain. The primitives for expression typically are class attributes or relationships. They include information on their meaning and the logical constraints of their application [SMM16]. Ontology is a powerful tool used with standard knowledge representation such as terminology, taxonomy, groups, individuals, and annotations [SSQ+19]. The function of an ontology is to act like a human brain. It operates with thoughts and relationships between several entities. This is seen as the way people perceive interconnected concepts [Ont18]. Implementing hundreds or thousands of requirements is difficult, as it takes a long time and is complex. In reducing query complexity, the architecture of the ontologies plays a significant role [CC10]. The vehicular data can be derived from multiple dimensions, such as the following:

- **Assets:** assets in a vehicle are known as information, machinery, element or a physical object or a logical object [AMS19]. The modern vehicles contain a wide range of ambiguous units and divers of communication protocols.

- **Potential Threats:** threats refer to the potential events of vulnerabilities when it is exploited by attackers [RAP19]. During the risk assessment, security threats must be viewed in the course of a threat analysis. Where these threats have been detected, a vulnerability analysis should be performed to specific security requirements [SSMG16]. In the vehicular domain, the threat analysis will play an integral role in identifying potential negative actions affecting the vehicle security. The threat analysis method can be divided into the following fundamental actions [STH+19]:

    1. Design the vehicle components with all the necessary information and assumptions related to security,
    2. Model potential opponents with the qualifications, actions, tactics, techniques, and procedures,
    3. Identify potential threats, apply the risk model to the process model, which is defined in step (1) and (2),
    4. Assess all threats that are detected to define the exact severity risk level,
    5. Select the security countermeasures to update the system design, to minimize the risk, and
    6. Detect missed or new threats, repeat step (3).

- **Vulnerabilities:** Security weaknesses are always present in computing and network systems. The attacker aims to manipulate the existing weaknesses to make it easier to access the system [Nor16]. Security weaknesses are always present in computing and network systems. The attacker aims to manipulate the existing weaknesses to make it easier to access the system [Nor16].
- **Exploits:** Finding and manipulating security weaknesses to access systems is both art and science. It is considered a game between security professionals and attackers [Nor16]. Exploitation is performed by attackers that can exploit vulnerabilities for malicious activity [RAP19].
- **Security Requirements:** In the system development process, choosing security countermeasures should be seen as a significant task. In order to keep the overall risk at an appropriate level of security. Furthermore, the identified potential threats and security vulnerabilities must be addressed by applicable security requirements. That is because of reducing the impact of the identified risk from a critical or a high level (i.e., unacceptable risk level) to a low-risk level (i.e., acceptable risk level with low impact) [AMS19].

3.5.2 Ontology Knowledge-Base Representation

The core of this work is to create a semantic description of the vehicular data. The data is presented in sets of vocabularies, taxonomies, and ontologies. That data contains a set of defined terms which are essential to how information is expressed [HFBPL09].

Vocabulary

It is a simple, well-defined set of terms define meaning in all contexts [HFBPL09]. Vocabulary is used to identify the terms used for a specific function, to explain reasonable relationships, and to establish relevant constraints using these terms [W3C19].

Taxonomy

The Ref. [vRB16] defines the multi-layers of a cybersecurity framework to provide the appropriate level of protection of vehicles. This approach implements a defence technique, believing hackers can gain access across individual layers [vRB16]. The taxonomies in this proposed work describe vehicles' terminologies according to a set of classes and subclasses as multi-layers for defining several object properties that define the relationships between different entities in the ontology structure. The taxonomy is designed to semantically identify all data in the vehicle used during the validation process, as will be discussed later. This design contains a hierarchy of data knowledge of classes and subclasses representing the main entities that are required in the security validation process (as is depicted in Figure 3.2 and will be discussed later in Section "Design of the Ontology Structure"). These classes define vehicle data in related sub-categories of classification schemes.

Figure 3.3 illustrates the vehicular component layers. The class of components classifies vehicle data into six categories. Each category accommodates vehicular components related to the matched type. The risk class also contains four risk levels (i.e., low, medium, high, and critical) that define the potential exiting threats' risk level. These levels measure a specific threat's severity level. Estimating the risk levels is necessary to determine security requirements that are needed to address identified threats. Vehicle sub-components schema looks like separate component layers

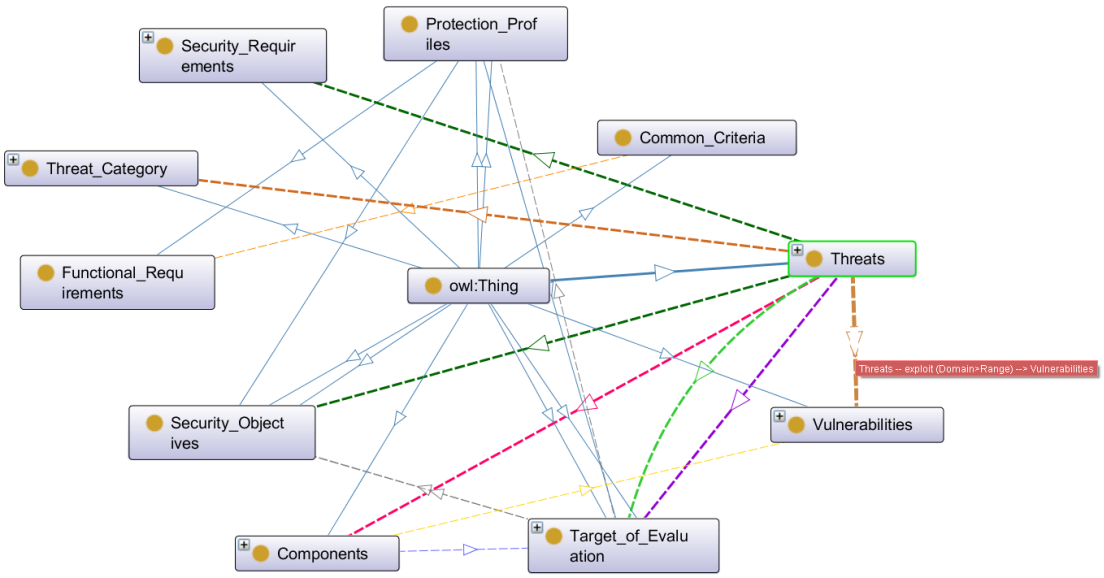


Figure 3.2: Ontology graph of classes and subclasses of the proposed cybersecurity framework

category; each layer contains components with common component type. In addition, each layer includes four sub-layers of risk levels to define the exact level of security related to each defined threat related to vehicle components.

This taxonomic structure better understands the ontological hierarchy (vehicular units) than defines all the ontology individuals in one main class. However, from the algorithm performance point of view, not much difference is noticed between identifying all entities as individuals to similar class (Components) or assigning into separate subclasses. Figure 3.4 demonstrates the retrieval time of individuals in one main class or subclass.

This test is performed using SQWRL. The SQWRL uses a default Semantic Web Rule Language (SWRL) rules and uses it as a template specification for a query [OD09]. Nevertheless, other studies use SPARQL language to test performance in the querying process. There is a notable difference in SPARQL's performance compared to SQWRL, where SPARQL is faster than SQWRL. In SPARQL, the inferred model is calculated one time only before the matching begins. Therefore, during the SPARQL execution time, the inference time is not considered. In the SQWRL scenario, the inference time influences the execution time [APM11].

Ontology

Building a correct ontology framework to align ontologies needs definition of the internal relationships between classes. In addition, it is essential to understand the relationship among other ontologies. The relationship within ontology classes is simple and straightforward; where the relationship across ontologies is less obvious [Gra08]. The ontology linking, ontology mapping, and ontology importing are three ontology combination paradigms that are identified to differentiate the relationship between the ontologies [HS10].

Ontology linking: links are used to describe interconnections between individuals from distinct ontologies. The ontology domain shall be a disjoint constraint to describe the ontology

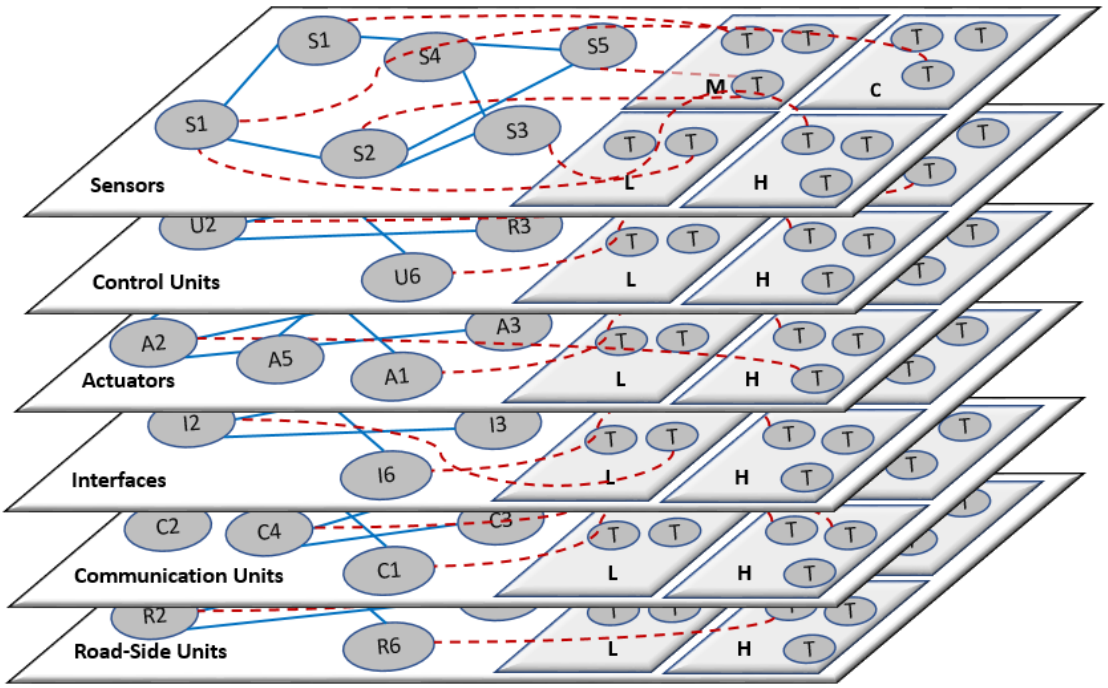


Figure 3.3: The taxonomy of vehicle components

relationship [HS10]. For example, in this work, the threat individuals are defined as an instance of the "Threats" class instances. The security requirement individuals are defined as "Security-Requirements" class instances. The relationship between these two classes is defined as "disjoint", where the OWL property is established as *OWL:disjointWith*. That means an instance at any of these classes is not be a part of the other one. Figure 3.5 illustrates how this work uses the ontology linking paradigm to represent the relationship between the "Threats" and "Security-Requirements" classes.

The disjoint relationship is stated to distinguish between predefined class instances. A simple OWL code (i.e.,Listing 3.1) shows how disjoint relationship is described.

```
1  <!-- http://www.semanticweb.org/shaabana/OnSecta.owl#
     Security_Requirements -->
2        <owl:Class rdf:about="http://www.semanticweb.org/shaabana/
     OnSecta.owl">
3       <owl:disjointWith rdf:resource="http://www.semanticweb.org/
     shaabana/OnSecta.owl#Threats"/>
4     </owl:Class>
```

Listing 3.1: A simple OWL language representing the disjoint relationship between threats and security_requirements

Ontology Mapping: This allows connecting the same ontologies on the same domain or a partly overlapping domain. The mapping paradigm is established to show how the entities are associated semantically, even when they are predefined from different ontologies. The mapping

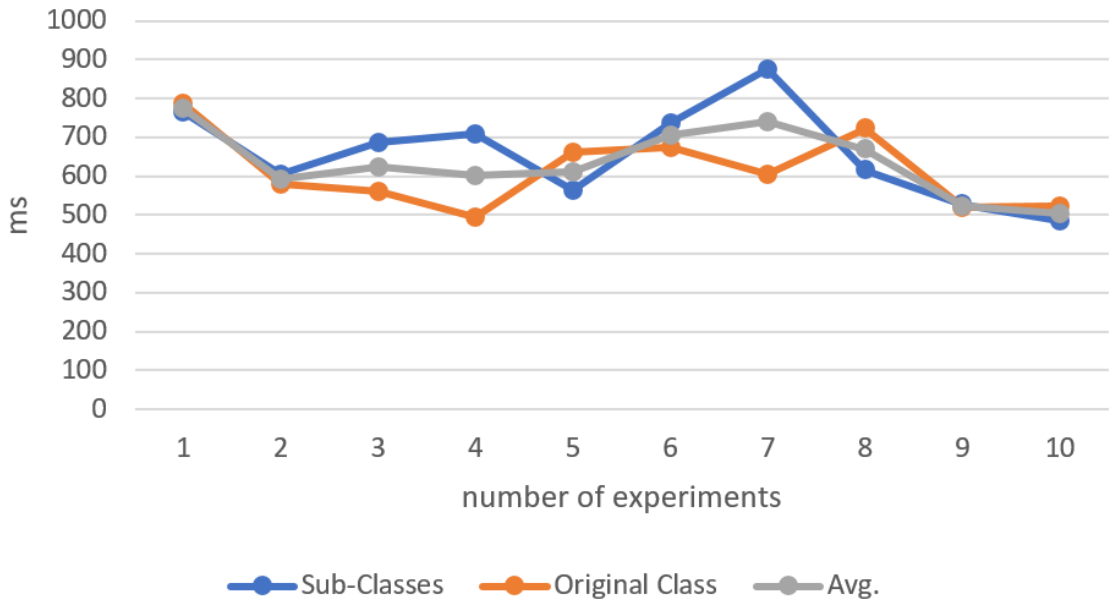


Figure 3.4: Querying performance according to the specified taxonomy

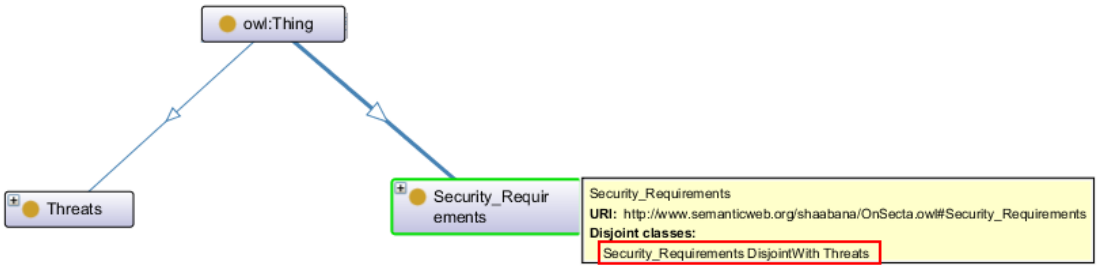


Figure 3.5: The ontology linking paradigm between "Threats" and "Security_Requirements" classes

enables for the resolution of the semantic heterogeneity between ontologies that are represented from the same domain but from different perspectives [HS10]. For example, as is shown later in this chapter, this thesis applies two varieties of security requirements. The first type is extracted from the security standard, where the second is collected from a protection profile based on common criteria. Such two forms are described in an ontology construction, each having the same domain and matched semantically, but each has a heterogeneous structure. These two ontologies are combined as subClass of the "Security-Requirements". In this work, Figure 3.6 illustrates how security requirements from the same entities but with alternative structures.

The figure illustrates security requirements hierarchy based on the protection profile and foundational requirements.

Ontology Importing: This allows a subset of individuals that are identified in one ontology to be imported into another. The importing paradigm is concerned with defining the semantic meanings and relationships specified in the original ontology to be used in the imported onto-

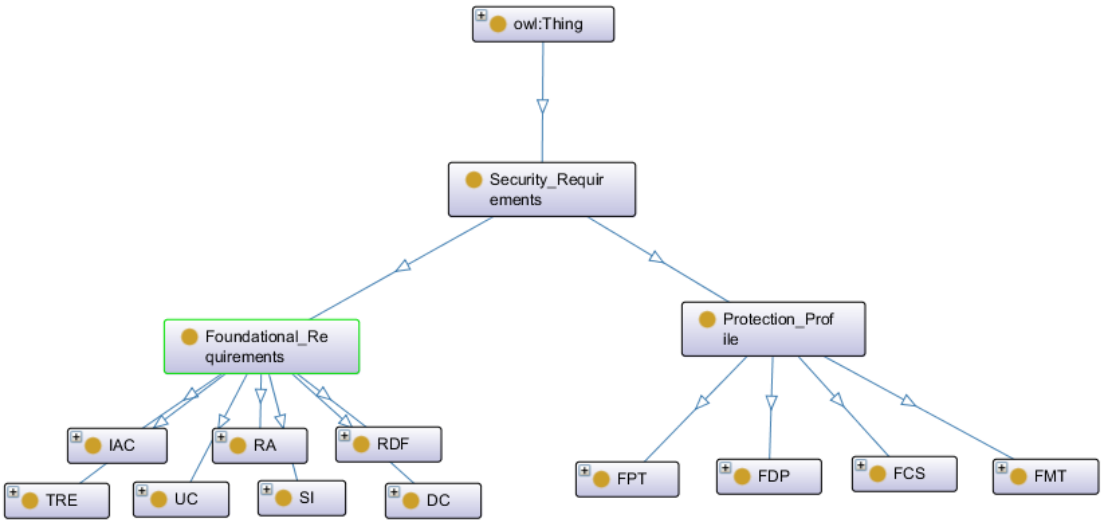


Figure 3.6: The ontology mapping paradigm between security requirements from the same domain and different structures

logy [HS10]. Within this work, this paradigm is integrated to import additional entities into the same ontology structure. For example, the security requirement individuals have collected to create the "Security-Requirements" class structure. As the work continues, a new group of individuals is introduced to implement the full set of security specifications used in this work.

3.6 Design of the Ontology Structure

The ontology determines a collection of primitive members to model a knowledge domain. The primitives of representation usually are classes, attributes, or relations. These include information on their meaning and restrictions on their consistent logical application. An ontology, therefore, represents the knowledge of a particular domain [SMM16]. The cybersecurity framework establishes multiple classes, categories, entities, properties, and annotations for all vehicle parts to be used in the verification and validation process. The ontology aims to explore the connections between vehicles, equipment, possible threats, and protections capable of addressing actual and anticipate security vulnerabilities.

Ontology is an aspect of semantic technology, as part of the W3C [3] semantic web standard stack [Ont18]. The semantic web offers a robust and practical approach to the multiplicity of information and information services. In order to make the World Wide Web understandable, semantic web technologies are created. Web resources are distributed inherently, and the descriptions of resources contained on the Semantic Web are thus also distributed. The ontology is made up of statements describing definitions, relationships, and limitations. It is similar to a schema in a database or a hierarchy diagram of object orientation. For areas such as finance and medicine, an ontology can capture depth for representing similar objects. A sufficient ontology facilitates cooperation between applications in the sense of ontology [HFBPL09]. Some of the ontology's essential features are to ensure that knowledge is commonly understood and to make explicit domain inference. This makes it essential to meet the challenges of collecting and querying data

[3] https://www.w3.org

in large organizations because the paradigm is integrated and interoperable [Ont18]. The ontology data is based primarily on a compilation of three complementary languages (i.e., RDF, RDFS, and OWL).

The ontology is considered the pillar of this proposed framework. It is used to create a series of essential definitions of the main problem and related solution that able to tackle the security weaknesses in vehicular domain [SSQ+19]. In this model, several classes are defined to formulate a domain integration between different vehicle entities and describe the proper relationship between them. An overview of all specified classes and their properties in this work is depicted in Figure 3.7.

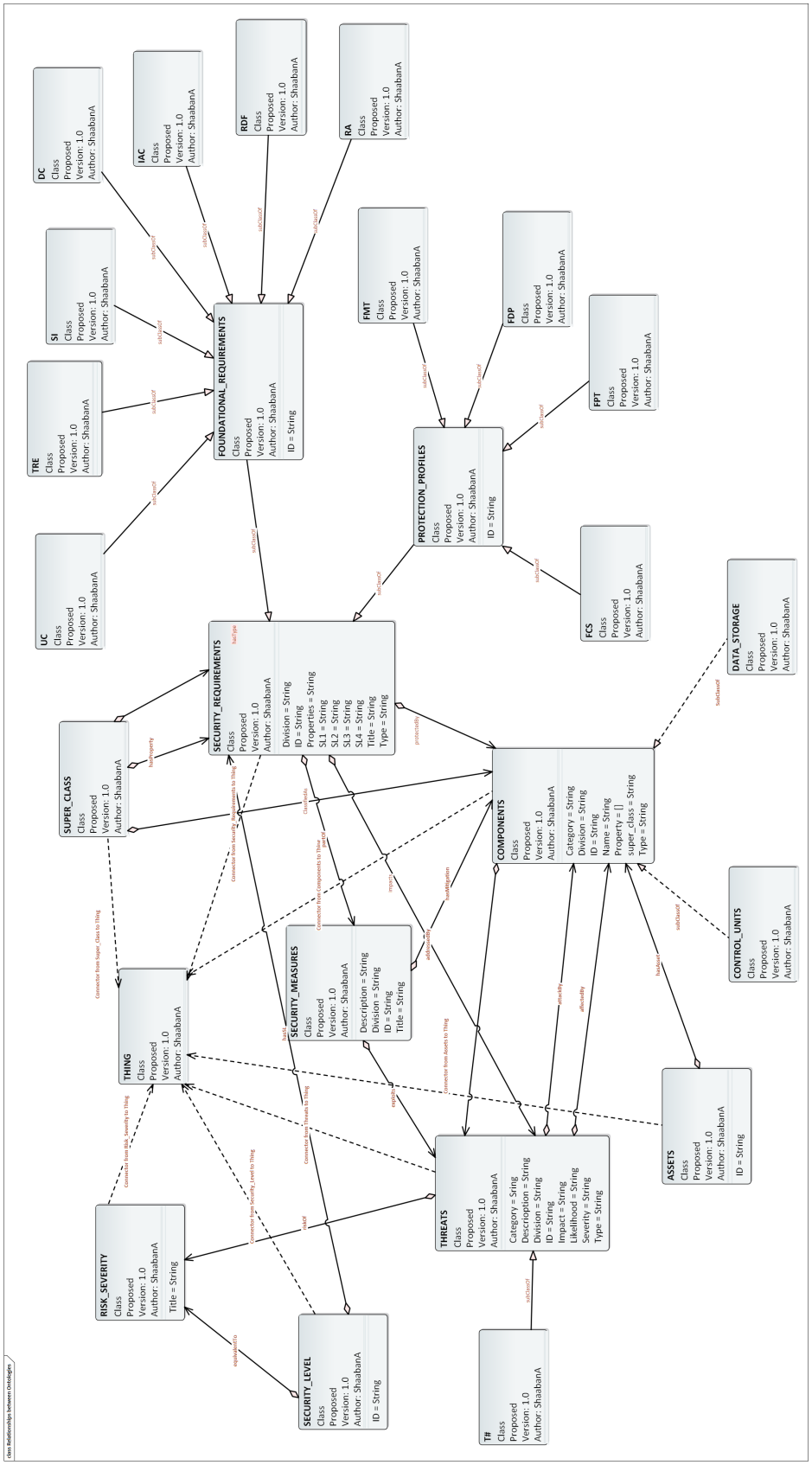


Figure 3.7: The class diagram of the metamodel of the proposed ontology structure

The figure shows various ontology classes, sharing similar features with instances. For example, the class "Threats" is created to include a collection of instances (i.e., threats) sharing similar properties. There are two types of properties in ontology; the first type is an object property that defines individual relationships, where the second type is data property, which defines links to literal values [HFBPL09]. As shown in the figure, each class has alternative connections with other classes reflecting on their relationships. On the other hand, each class includes a set of data properties created to describe literal values. In this work, the string data type is commonly used to define the type of data properties and represent the relationship between individuals and literal values. For example, all individuals in the "Threats" class have a hierarchy of tagged values representing data values such as ID, title, impact, likelihood, risk, severity, etc. Some of the selected data properties are described as follows:

- **ID:String-** a unique threat identification value,
- **Title:String** specifies the threat name,
- **Impact:String** represent the value of the severe effects of the threat in case of it is triggered,
- **Likelihood:String** represents the probability of risk occurrence,
- **Risk:String** Specify the value of the risk assessment process to determine the risk level, and
- **Severity:String** This value shows the severity degree of the risk; different parameters estimate this value, such as low, medium, high, or critical.

Figure 3.8 illustrates the object properties between threats and other classes such as Security_Requiremnets, Security_Measures, Components, and Risk_Severity. These properties define relationships between individuals as instances of "Threats" class with individuals in other ones. There are different object properties created for building a logical knowledge representation between these entities.

- ***attackBy:*** This property determines the relationship between a component and a potential threat. If any of a component's security measures are exploited, this component could be attacked by at least one threat. Consequently, the aggregation relation is applied here to express the meaning of this relationship. In ontology, the relationship is defined as annotations. The annotation are statement or triples (i.e., <*subject* >, <*predicate* >, and <*object* >), which have properties for predicate. The relationships have to be defined between two arguments, the first is called the domain, and the second is called range. The domain sets the combination of subject individuals (resource) to the statement using the described property, where the range defines the combination of individuals that are object (properties) to the statement using the described property [HFBPL09]. Therefore, the whole statement determines as follows:

$$Componnets \overset{attackBy}{\longrightarrow} \text{Threats}$$

- ***affectedBy:*** Is another type of a relation between threats and components. The semantic of this relation is defined to show the type of threat that could affect particular components. This relationship aggregates all the affected components together with the potential threat(s).

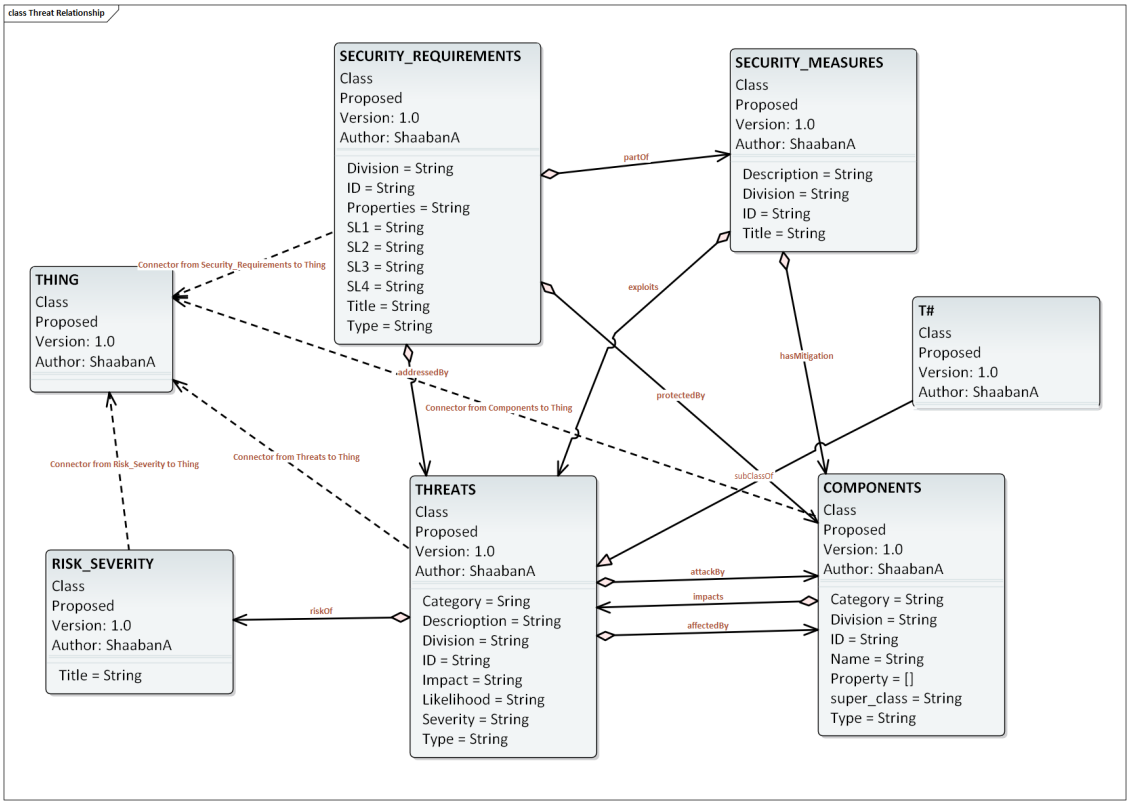


Figure 3.8: Data properties and object properties of the "Threats" class

- ***impacts:*** Impact is another relationship defined in ontology to present the threat's impact on a component. The threat may aim to compromise confidentiality, integrity, and availability. The domain here is the threat, where the range is the component, which the whole statement (triple) of this relationship defines as follows:

$$Threats \overset{impacts}{\longrightarrow} \text{Components}$$

- ***riskOf:*** Each threat has a different risk severity level, which defines the damage level that may affect the component when this risk happens. The risk evaluation is not detailed in this thesis study. However, later in the chapter, we will discuss how to evaluate the severity level of threats to match these values with the related security levels of the security requirements. The risk evaluation or risk assessment process is a systematic approach of identifying and analyzing the hazards (i.e., safety) or threats (i.e., security) and estimating a level of risk severity for each hazard or threat [RPMS17]. This activity is based on the parameters of impact and likelihood used to evaluate the specific risk level. On the first hand, it is essential to ensure that different impacts do not damage the vehicle or cause other accident scenarios. As discussed by [SLAMH18], there are four levels of impacts in the automotive domain:
  - causes immediate damage to the environment or human lives **Safety (S)**,
  - causes the loss of control over personal information **Privacy (P)**,
  - causes financial damage **Financial (F)**, and
  - negatively impacts the operation and traffic flow **Operation (O)**.

Five alternative factors are discussed in [ITPAO13], which are derived from the common methodology for information technology security evaluation, version 3.1 [Cri17]. A range of numerical values express these factors to define the level of attack scenarios to exploit vulnerabilities. The factors of attack potential are described as follows [ITPAO13]:

- **Elapsed Time:** the time taken to exploit and identify vulnerabilities,
- **Specialist Expertise:** the required technical knowledge for the attack,
- **Knowledge of Evaluation Target:** this represents the knowledge of the system design and operation,
- **Window of Opportunity:** refers to the amount of access to the target unit, and
- **Equipment:** IT hardware, software, or other equipment required for exploiting vulnerabilities.

Later, the severity of each of the detected potential threats is evaluated based on parameter values of the likelihood and impact level; this part will be discussed in more detail in Chapter 6.

- ***exploits:*** Reflect the relationship between the applied security measures and threats. The threat itself involves exploitation, as it is the way attackers can start malicious actions [Rap20]. Components, security requirements, and threats typically describe security measures, which are common in these classes. The security measures can be the system's security weakness, which could be exploited by threat(s) to open the way to the attacker into the malicious goal. For example, a threat T exploits a security vulnerability V in an asset A to target the vehicle system; the vulnerabilities may be described as any security properties are added to a vehicle asset to defend the asset from cyber attacks. However, these security properties may be considered weak points that attackers can start performing malicious actions, such as implementing weak cryptographic algorithms or using a short encryption key.

- ***hasMitigation:*** As discussed above, each component has a set of security specifications to reduce the risk. Applying security mitigation is the standard method of avoiding/reducing attackers' activities. The "Security Measures" class describes a collection of such mitigation measures as part of the asset to protect it from malicious activities. Therefore, the "hasMitigation" relationship defines the connection between the component and the applied security specifications from the "Security Measures" class.

- ***protectedBy:*** This property is used to define the component-security relationship, which implements a security mitigation set that can reduce risk from alternative cyber attacks.

Sensitive data is considered a valuable asset requiring more security concern [ISO12]. Assets are also concerned in this work as a related unit to particular components. The asset is a vehicle considered as data, device, component, or either a physical or a logical object. The asset identification process concerns with the following tasks [Ass13]:

- create an asset record,
- identify asset information,
- define the topological structure of interconnected assets.

The identification of assets in the Information Security Management System (ISMS) should be identified and followed by a given point, as discussed in ISO/IEC 27003 as follows [iso10]:

- unique name,
- process description,
- process critically,
- owner of the process,
- inputs and outputs from the process,
- classification of information (i.e., confidentiality, access control, non-repudiation, and other properties).

The assets in this research are described as a part of a component that could have high impact, threatening either the component or the entire system.Figure 3.9 illustrates the relationship between the vehicular components and correlated assets. **hasAsset** is used to represent the connection between assets and components within the vehicular system. Each component could contain a particular unit, data, or an object, which is considered the attackers' main target.

The domain here is a component, where the range is an asset; this annotation is described as follows:

$$Components \xrightarrow{hasAsset} \text{Assets}$$

According to the above relationships between components, threats, and security measures, Figure 3.10 illustrates an example that is used in this work demonstrating these relationships between alternatives entities within the vehicular domain. The vehicular units contain a set of critical assets that could be the real target of a cyberattack, such as "Cryptographic Keys" asset of the V2X HSM (Vehicle-to-anything Hardware Security Module) component as discussed in [Car19] by "Car 2 Car Communication Consortium" [4].

A collection of assets is described as representing a specific component. Therefore, the asset-component relationship is known as an aggregation relationship. The class "V2_XHSM" contains a set of assets defined as individuals of this class. The "hasAsset" property is created to describe class-correlated individuals. Therefore, "V2X_HSM" has "Cryptographic_Keys" The hasMitigation relationship is the security property applied to the asset to protect from a potential threat. However, a potential threat could exploit the same security property, as described in the example. The affectedBy and attackBy are the relationships between the asset and the threat based on the security property violation. Therefore, this asset is affectedBy and attackBy "T13_1" when this threat exploits a vulnerability (i.e., confidentiality). In addition, this security property is specified as part of a security requirement "FCS_CKM_1_1" in the selected protection profile.

The security requirement class includes two main alternative resource data sets. As described in Figure 3.11, the first group is formed from a large set of security requirements from IEC 62443, which aims to be integrated into this work regarding the absence of a real automotive security requirement. The second group of security requirements is based on the protection profile [Car19]. The next section will discuss the structure and usage of these security requirements and how these two different data are used to identify appropriate vehicle security specifications to counter potential threats.

[4] https://www.car-2-car.org

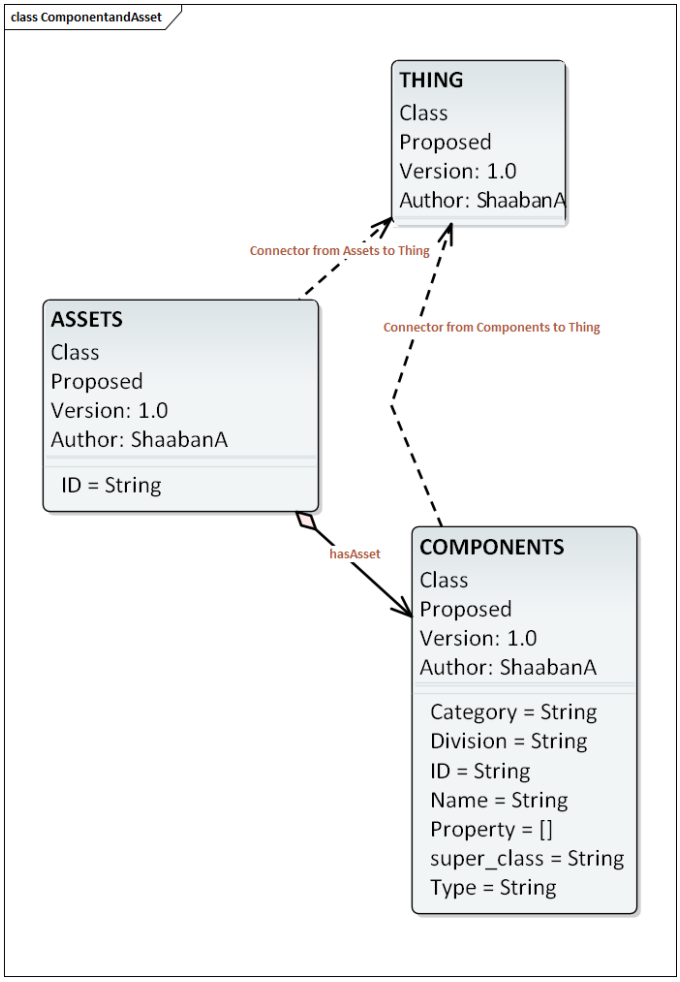


Figure 3.9: Component-asset correlation through the **hasAsset** relationship

Each security requirement described in the IEC 62443 security standard has different security levels called security capabilities (i.e., SL-C: represents the security level that the system's components or the whole system can achieve without extra measures) [isa13]. In this work, we use these levels to map which risk severity levels are estimated according to specific malicious behaviour of a potential threat. Therefore, the Security_Requirements class has a relationship with the Security_Level class is defined as "hasSL" property. As discussed in the IEC 62443-4-2 [IEC19], the IEC 62443-1-1 defines the security requirements into seven categories of security requirements, called foundational requirements (FRs). The Foundational_Requirement class is created to classify the security requirements as per the primary security standard. Hence, the Foundational_Requirements class is classified as subclassOf the Security_Requirements main class. The proposed security ontology structure aims to be more flexible to accommodate a wide range of security requirements from multiple resources. This research work utilizes a new set of security requirements based on a particular vehicular asset based on a protection profile. These selected security requirements are defined according to the common criteria. Therefore, the Protection_Profiles class is created and integrated within this work to use other security requirements from multiple protection profiles. Each security requirement contains a set of

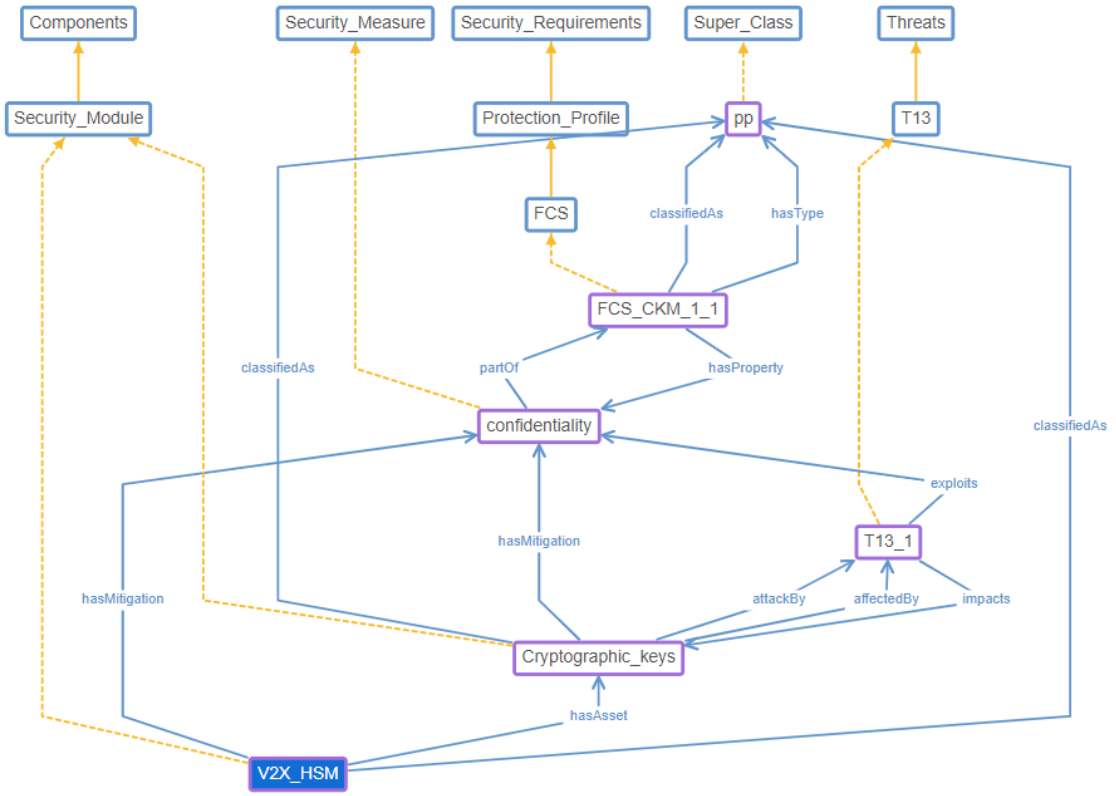


Figure 3.10: A Part of the ontology taxonomy of components, assets, properties, and threats

security measures that are considered part of the content security requirements. Therefore, partOf relationship is defined between the Security_Requirements and Security_Measures classes to express that the security measures can be identified from one or multiple security requirements.

3.7 Verification and Validation Process

As discussed earlier, the vehicular content (i.e., components, assets, threats, security measures, and security requirements) are used to build a complete ontology structure. The ontology improves the accuracy of the verification and validation process by creating and executing a set of logical rules for inferring a set of applicable security requirements for particular security weaknesses. The inferred results are compared to the selected security requirements using verification and validation processes for checking the completeness and correctness of the selected security requirements against the existing security weaknesses. The discovered security issues are handled by integrating the inferred security requirements within the ontology model to address the existing security issues and protect the affected vehicular's components/assets. The proposed framework generates a set of test-cases according to the existing vehicular information and the obtained results to define and demonstrate the vehicular components/assets compliance with a particular level of prerequisite security requirements. Figure 3.12 depicts a preliminary design of this process to give a simple conceptual design of the verification and validation process in this reach work.

The theorem proving the verification method is applied and integrated into this work to verify the correctness of the ontology structure's consistency and assure the ontology hierarchy's com-

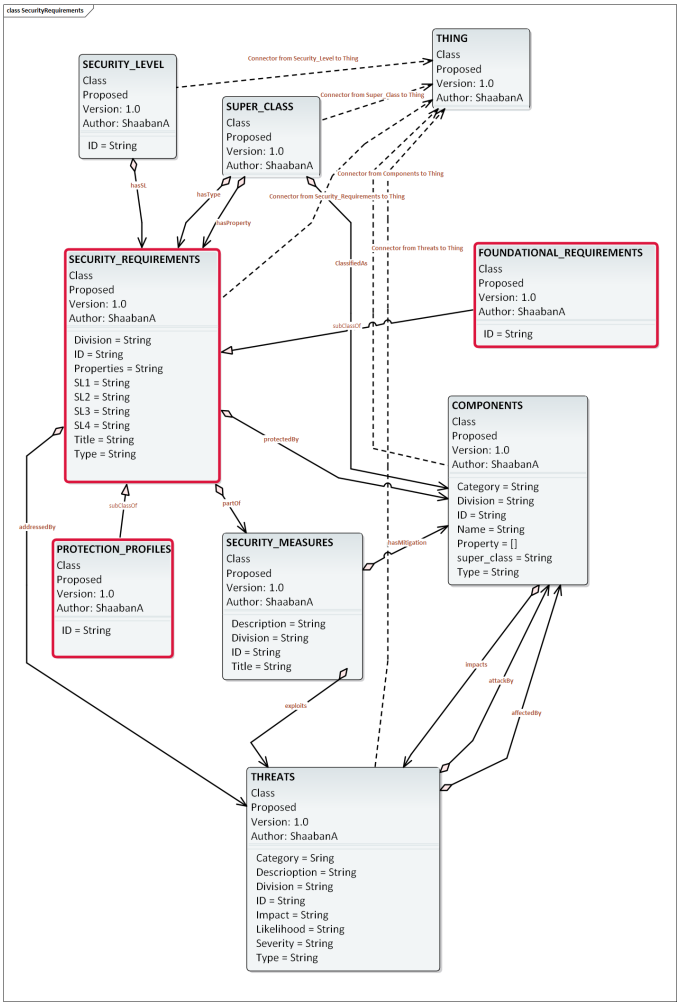


Figure 3.11: The relationship between the protection profiles, security standards, and the security requirements class

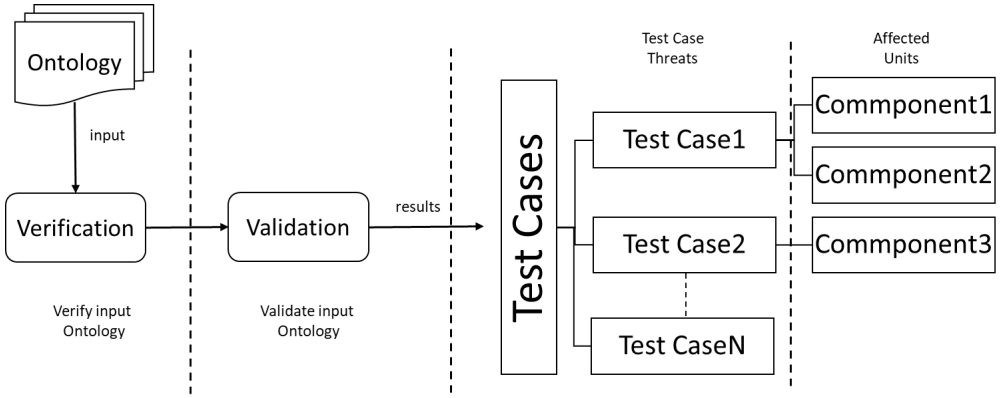


Figure 3.12: The conceptual design of the verification and validation processes

pleteness with existing relationships. Formal verification uses logical methods to build complete,

precise mathematical terms. The formal verification describes the hardware and software of a system in a mathematical manner [AdMK17]. However, the main issue is which mathematical formalization should be used. Ref [Har08], define some common formalisms, such as:

- **Propositional logic:** the mathematical formulas are defined from variables, constants, and connectivities.
- **Full first-order logic:** expressing objects from a set of variables and constants.
- **Higher-order logic:** a form of the logical expression, which can define many orders than the first-order logic.

The propositional logic could be defined is the closest type that followed in this work to verify the ontology individuals' correctness. The SQWRL and SWRL languages used in this work are based on Horn logic. Horn logic is considered as a formal language to express different rules for using in logic programming. These rules are used to compute a set of answers using inference procedures [WRWW05]. The first and high order logic are also used to express the relationship between the domain and range of some ontology entities, as discussed in Chapter 6.

The validation process plays an essential role in the vehicle development life-cycle to determine the correctness of the applied security requirements against the existing security issues. In this work, we use the testbed execution method, because this is the proposed method based on the ISO 27005 [iso11] according to the ISO 31000 risk management process. This process consists of multiple activities applied to each vehicle component/asset individually, which could be defined as a unit testing approach for examining the effectiveness of the applied security requirements for its relevant vehicular component/asset. This validation method will be applied to a considerable amount of vehicular data in an ontological form. Test-cases will be generated according to the identified/inferred threats with the vehicle model's relevance of the affected components/assets. More detailed information about the ontology-based cybersecurity framework's design is presented and discussed in Chapter 6.

3.8 Chapter Summary

This chapter discussed the main research strategy process followed in this work to define research artefacts for creating new innovative solutions for addressing the cybersecurity issues in the automotive domain. The next chapter discusses the methodology for adapting the collected data in this research and building the vehicle model's ontology hierarchy according to the previously discussed structure.

The next chapter includes more details about the collected data used in this research. It discusses the sources of data and clarifies the steps followed to adapt these data into the context of this work for building the ontology structure of a vehicle model.

4 Research Data Collection: Security Requirements

This chapter mainly focuses on the security requirements integrated and incorporated within this study for developing a cybersecurity framework for the automotive domain to ensure the correctness of the applied security requirements. The different resources used to gather the various security requirements are outlined here before discussing how collected contents of security requirements are translated into an ontology representation.

This chapter includes content from some articles published within this thesis work. The following list refers to the selected publications that are integrated within the context of the chapter:

- *Schmittner, Christoph, Abdelkader Magdy Shaaban, and Johannes Hellrich. 2019. "Automating the Construction of a Security Threat and Mitigation Pattern Library." in INCOSE Artificial Intelligence for Systems Engineering.*
- *Shaaban, Abdelkader Magdy, Erwin Kristen, and Christoph Schmittner. 2018. "Application of IEC 62443 for IoT Components." Pp. 214–23 in Computer Safety, Reliability, and Security, Lecture Notes in Computer Science, edited by B. Gallina, A. Skavhaug, E. Schoitsch, and F. Bitsch. Cham: Springer International Publishing.*

4.1 Data Sources

The security requirements selected in this work are based on alternative data resources. It is essential to select reliable data with a high level of trustworthiness in order to validate the work's efficacy. Therefore, this work uses IEC 62443 [ISA18], V2X Hardware Security Module Protection Profile [Car19], [Car18], and common criteria [ISO09b].

4.1.1 IEC 62443 Security Series

Cyber-Physical Production Systems (CPPS) represent one of the driving forces in technology today. CPPS integrates and builds on a variety of existing technologies and components, such as robotics, industrial automation and control, Internet of Things (IoT), big data, and cloud computing [MHSP17]. The Internet of Things (IoT) consists of small, widely distributed components which offer many benefits for industrial systems. IoT is a new enhancement for Industrial Automation Control Systems (IACS), where billions of devices with smart data processing capabilities are networked via the Internet [SKE17]. In former days, the corporate networks for production planning, work scheduling, material ordering and other administrative work were isolated from the Industrial Automation Control Systems (IACS) networks. This top-level enterprise information technology infrastructure was already exposed to external attacks due to necessary communication links to the outside world. As corporate networks are covered by IACS networks, and the proprietary IACS networks were replaced with commercial-off-the-shelf equipment using Ethernet-TCP/IP

technology, the same cyber vulnerabilities threaten the IACS networks. Additionally, overall system complexity has dramatically increased, and new threat sources now jeopardize the controllers in the IACS networks, for which they were never designed. Replacing them with attack-proof controllers is in many cases not possible or not offered by the machinery supplier. This would also disturb the continuous operation of the production process.

Next-generation IACS, which already exist in various implementation stages, are more open architectures that are no longer limited to the plant area. Communication links, whether wired or wireless, connect the components of different operational processes. There is no need to place the components as near as possible for short interconnections. On the contrary, these network interconnection arrangements allow the total breakup of conventional enterprise architectures. For the following example, see the well-known Purdue Model of a network layered enterprise architecture, shown in Figure 4.1. The name Purdue Model [Wil94] came from Purdue University, Indiana, U.S, where Theodore J. Williams developed the "Purdue Enterprise Reference Architecture (PERA)" in the 1990s. Enterprise Resource Planning (ERP), the top levels of the Perdue model, is defined for most industrial plants. The same holds for the next layer, the Manufacturing Execution System (MES). Level 2, the Supervisory Control and Data Acquisition (SCADA) layer, is the next candidate to be moved partly or entirely outside the plant. In future, the levels 1 and 0 will be connected wireless in a so-called ad-hoc network, where the different components are connected among each other depending on the product which shall be produced [Ser20]. The increased interconnection and collaboration between manufacturing steps in the supply chain, multiple levels will be interconnected.

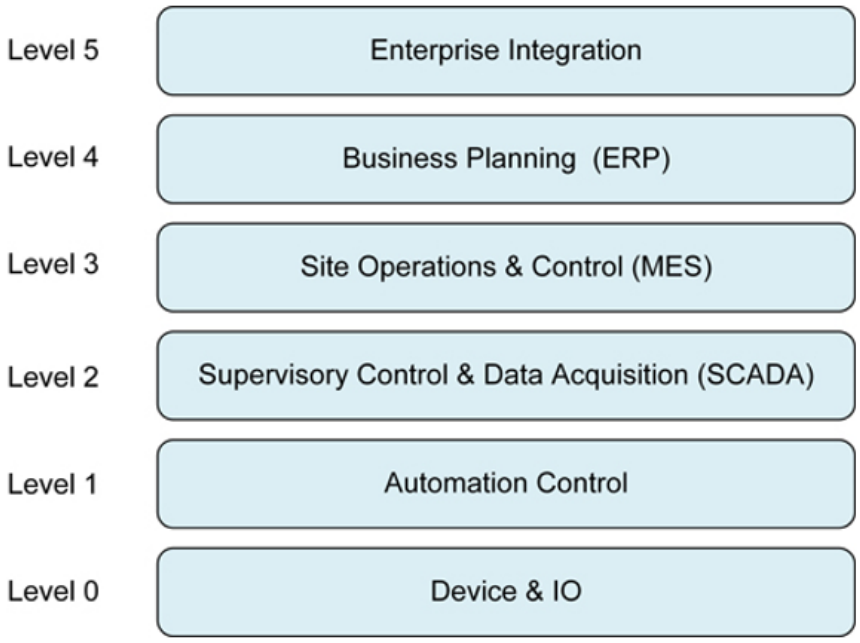


Figure 4.1: Network layered architecture (Purdue Model) [Ser20]

The IEC 62443 series's primary goal is to present a framework that addresses current and future security vulnerabilities in industrial systems. It enables security risk management for the complete life cycle and all layers of industrial networks [IEC19]. This is done by dividing the system into zones, defining security levels for each zone and specifying security capabilities that enable a

Table 4.1: The IEC 62443 security series [IEC19]

General	IEC 62443-1-1	IEC TR 62443-1-2	IEC 62443-1-3	IEC TR 62443-1-4	
	Concepts and models	Master glossary of terms and abbreviations	System security conformance metrics	IACS security lifecycle and use-cases	
Polices and Procedures	IEC 62443-2-1	IEC 62443-2-2	IEC TR 62443-2-3	IEC 62443-2-4	IEC TR 62443-2-5
	Security program requirements for IACS asset owners	IACS protection levels	Patch management n the IACS environment	Security program requirements for IACS service providers	Implementation guidance for IACS asset owners
System	IEC TR 62443-3-1	IEC 62443-3-2	IEC 62443-3-3		
	Security technologies for IACS	Security risk assessment and system design	System security requirements and security levels		
Component	IEC 62443-4-1	IEC 62443-4-2			
	Secure product development lifecycle requirements	Technical security requirements for IACS components			

component to be integrated into a system environment at a given security level (SL). The IEC 62443 standard consists of thirteen documents classified into four main groups: General, Policies and Procedures, System, and Component. Table 4.1 illustrates, a description of the IEC 62443 series according to [ISA18]. The first two groups represent concepts, uses cases, policies and procedures associated with ICS security. The other two groups, System and Component, define the technical requirements for networks and system components [ISA18].

General: the first category of this series is General, which involves discussion and subjects that are common throughout the entire series. The IEC TS 62443-1-1 represents the terminologies, principles, and models for IACS security. There are seven foundational requirements (FRs), as follows: [ene20]:

FR1:Identification and Authentication Control (IAC): The main objective of the identification and authentication control is to validate a user's identity before getting permission to access a system. This permission could be granted to individuals, processes, software, or hardware devices needing a permit to access a system [isa13]. This foundational requirement is essential in the automotive domain to define a robust authentication mechanism to validate the authentication of vehicular units, individuals and assets among others, as well as any other entities that request communication with the vehicle.

FR2:Use Control (UC): This foundational requirement restricts system access to authorized users only. It aims to prevent unauthorized system activities by checking the requested privilege before allowing any entities to start interacting with the system [isa13]. This is essential in the automotive domain to permit authorized entities from accessing internal vehicle assets such as critical data.

FR3:System Integrity (SI): Security integrity aims to prevent an authorized entity (i.e., individuals, processes, software, or hardware) from compromising any parts of the system [isa13]. These foundational requirements are essential in the vehicle domain to prevent unauthorized nodes within the vehicle or even outside the vehicle (e.g., remote attack) from manipulating, damaging, or compromising any assets in a vehicle, such as vehicular data.

FR4:Data Confidentiality (DC): Data confidentiality intends to prevent unauthorized disclosure activities of data on communication channels or data stores in repositories [isa13]. This category of security requirements is essential in the automotive domain to keep all transmission data flowing in or out of vehicles secret, and to avoid revealing transmission data by any illegal methods.

FR5:Restricted Data Flow (RDF): Restricting the unnecessary data flow by creating security boundaries called security zones and conduits for communication channels [isa13].

FR6:Timely Response to Events (TRE): Create notifications to respond to any malicious activities on a system [isa13]. This could be helpful in a vehicle domain; the vehicle can manage and generate automatic notifications to authorities in case of any cyber-attacks (or even reporting an accident) which may have happened to the vehicle.

FR7:Resource Availability (RA): Ensure the availability of a system against different types of denial of service attacks [isa13]. This could also be useful in the vehicular sector to ensure the response of a vehicle or any other road units (e. g., road-side units RSU) before establishing communication.

Polices and Procedures: This category is concerned with the security of the IACS, which provides security requirements to evaluate the protection level of operational IACS. The IEC 62443-2-1 describes the owner of the asset for IACS and outlines the requirements for creating and evolving the security program. This series specifies the necessary capabilities of the protection needed to ensure the operation of an IACS. The second set is IEC/IS 62443-2-2, which specifies a methodology and framework of an IACS for assessing defence according to the security level and implementation of the related processes. The next and third series in this category is IEC / TR 62443-2-3, which offers details about the exchanging format from assets owners to product suppliers. Additionally, it describes activities associated with the suppliers and deployment of the patches by asset owners. The last series of this category is the IEC 62443-2-4, which defines security requirements for IACS service providers. Such capabilities are specified in IEC 62443-3-3, which the service provider guarantees are to be maintained within the scope of Automation Solution [ene20].

System: The IEC 62443-3-1 standard provides an overview of the advantages and limitations of existing network security technologies. IEC 62443-3-2 standard addresses security risk assessment and network design. Finally, the IEC 62443-3-3 standard outlines general system security requirements, emphasizing that performance should not be jeopardized during the addressing process of these requirements [iII20].

Component: The Component group consists of two documents. The IEC 62443-4-1 standard defines the development process of ICS products to reduce the number of security vulnerabilities in control system solutions. The IEC 62443-4-2 standard specifies the technical requirements for securing the individual components of an ICS network. The standard documents are aligned with IACS life-cycle phases. Figure 4.2 shows the life-cycle phases of IACS cybersecurity [iII20].

In order to be successful in IACS cybersecurity, all target audiences (Owner, Integrator, Supplier) have “shared responsibility” for all phases of the IACS cybersecurity life cycle [Ris]. The IEC

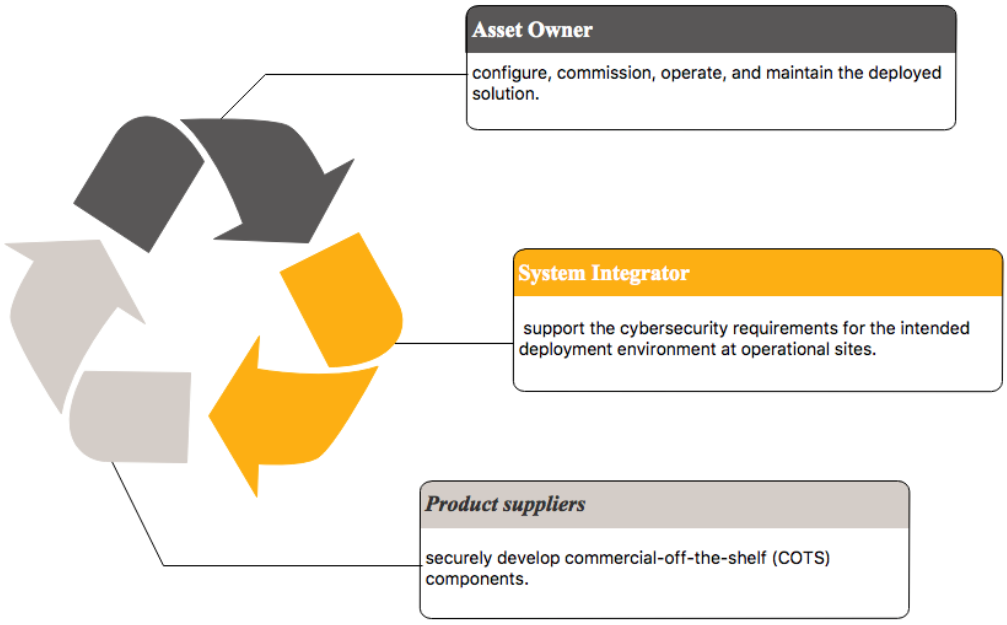


Figure 4.2: IACS cybersecurity life-cycle phases

62443 standard defines rules and methods to operate IACS networks by requirements, controls and best practices recommendations.

Zones and Conduits Concept

The IEC 62443 security standard focuses on the security analysis of the entire manufacturing plant. Based on a security analysis the plant is subdivided into zones which are called Security Zones. Zones are based on specific characteristics or security needs of the elements [SKS18b]. The standard describes zones and conduit requirements (ZCRs) for a system under consideration (SuC). The SuC means defining a collection of IACS and all related assets for performing risk analysis. The IEC 62443 describes the steps to establish zones, conduits, and their relationships to ZCRs; these steps are represented in a workflow diagram, as illustrated in Figure 4.3 [IEC15].

According to the IEC 62443-3-2 [IEC15], the ZCR's are described as follows:

- **ZCR1 - Identification of the SuC:** the SuC shall be identified with a clear definition of the security boundaries and identification of all access points to the SuC.
- **ZCR2 - High-level risk assessment:** performs a high-level risk assessment of SuC to identify unmitigated risks. This helps to gain a better understanding of the worst-case risk presented by SuC. The assessment helps facilitate asset grouping in various zones and conduits. The quantification of high-level risk is often accomplished when used as a risk matrix to define the relationship between likelihood and impact values. The likelihood and impact values have five levels of parameter values. The likelihood is defined as follows:
  - **Level 1**: Trivial
  - **Level 2**: Minor
  - **Level 3**: Moderate

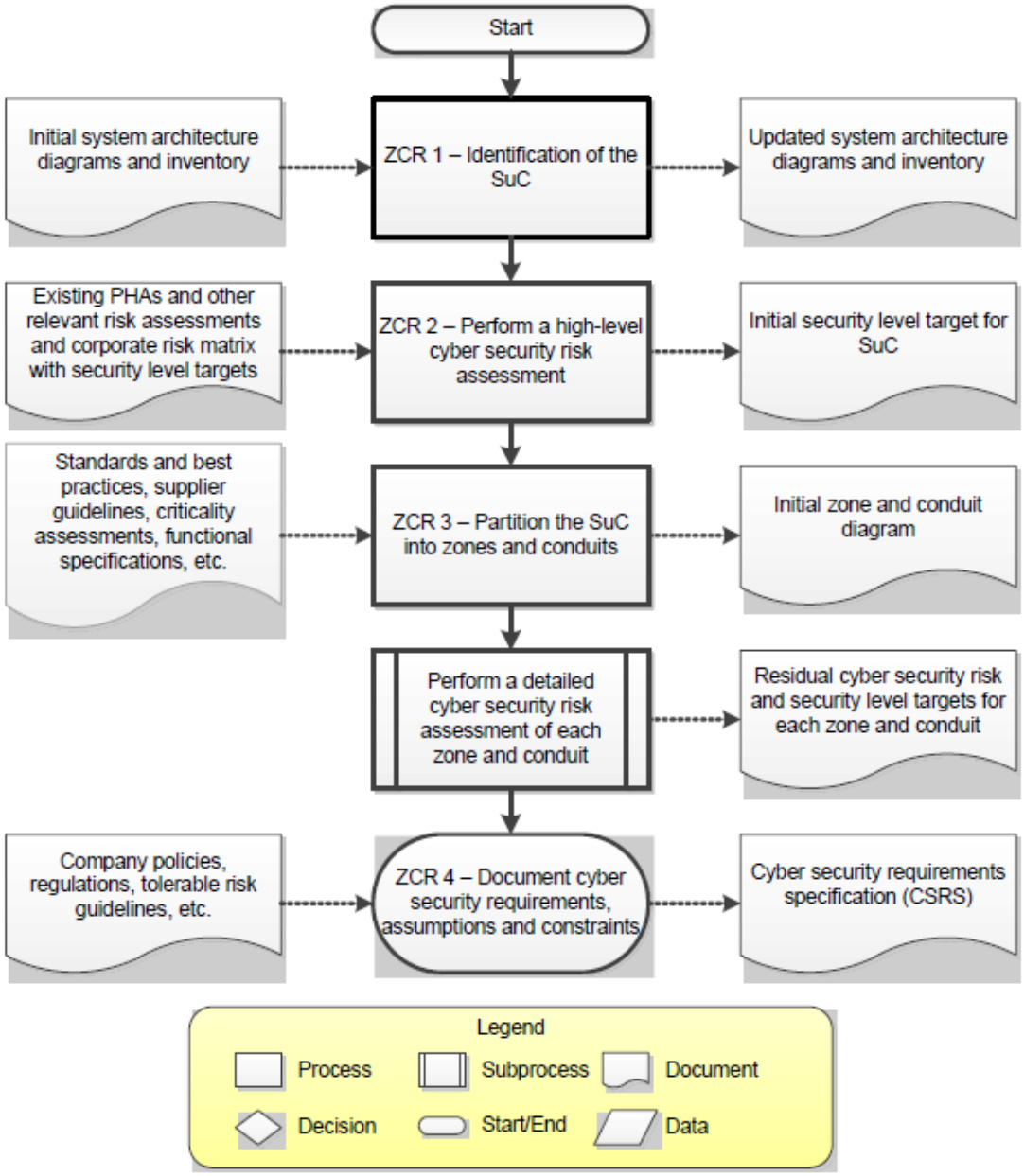


Figure 4.3: IEC 62443 workflow to establish zones and conduits, and their relationship [IEC15]

- **Level 4**: Major
- **Level 5**: Critical

The same as for the impact parameters:

- **Level 1**: Remote
- **Level 2**: Unlikely
- **Level 3**: Possible

- **Level 4**: Likely
- **Level 5**: Certain

- **ZCR 3 - Partition the SuC into zones and conduits:** this phase splits up the complex overall system into separate zones and conduits. Kloibhofer et al. [KKJ18], represents that the TrustworSys demonstrator [1], as an innovative configuration of smart product and their production in a smart factory. Figure 4.4 illustrates the TrustworSys demonstrator with separate zones and conduits.

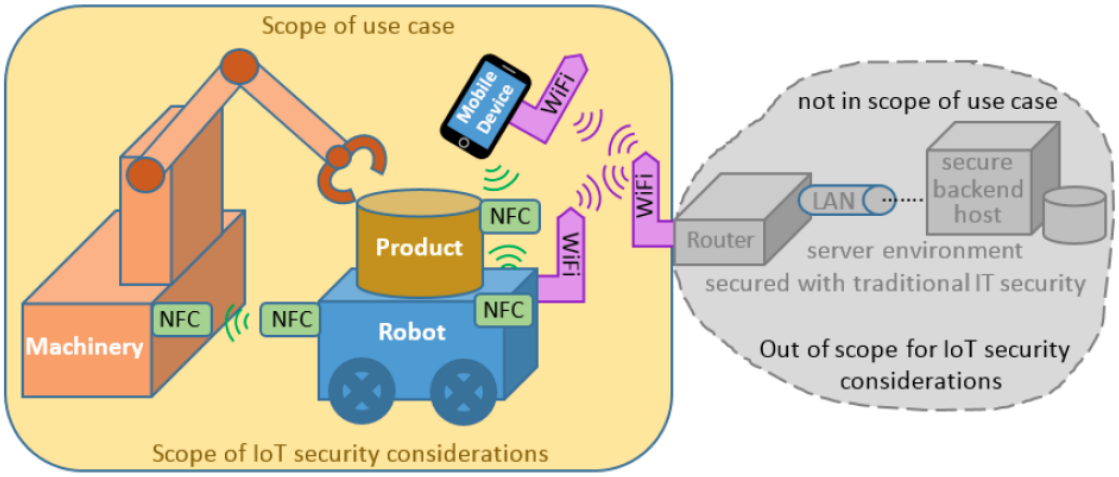


Figure 4.4: Example of SuC partitioning in zones and conduits [KKJ18]

The figure displays separate zones of the system, as presented in [KKJ18], it contains a set of nodes, such as follows:

- robot,
- product,
- machinery, and
- mobile configuration device.

Besides, the system defines two conduits, such as:

- NFC - near-field communication,
- WiFi communication.

The risk matrix is then established to define the relationship between the likelihood and impact, as discussed in IEC 62443-3-2 [IEC15]. The risk matrix explanation, likelihood, and impact clarification will be discussed in Chapter 6.

- **ZCR 4 – Document cyber security requirements, assumptions and constraints:** this is the last phase to assess the cyber risk for each zone and conduit to individually evaluated target security levels [KKJ18].

[1] http://www.iosense.eu/index.php/2018/11/20/intermediate-results-of-the-trustworsys-demonstrator/

Case-Study: Operating Plant

Shaaban at el. [SKS18b] defines an example of using the IEC 62443 security standard in an operating plant. Figure 4.5 shows an example of an operating plant used in the use case. The operating plant is divided into four security zones (Zone 1, Zone 2, Zone 3, and Zone 4). Three conduits (Conduit 1, Conduit 2, and Conduit 3) define the communication paths between these zones.

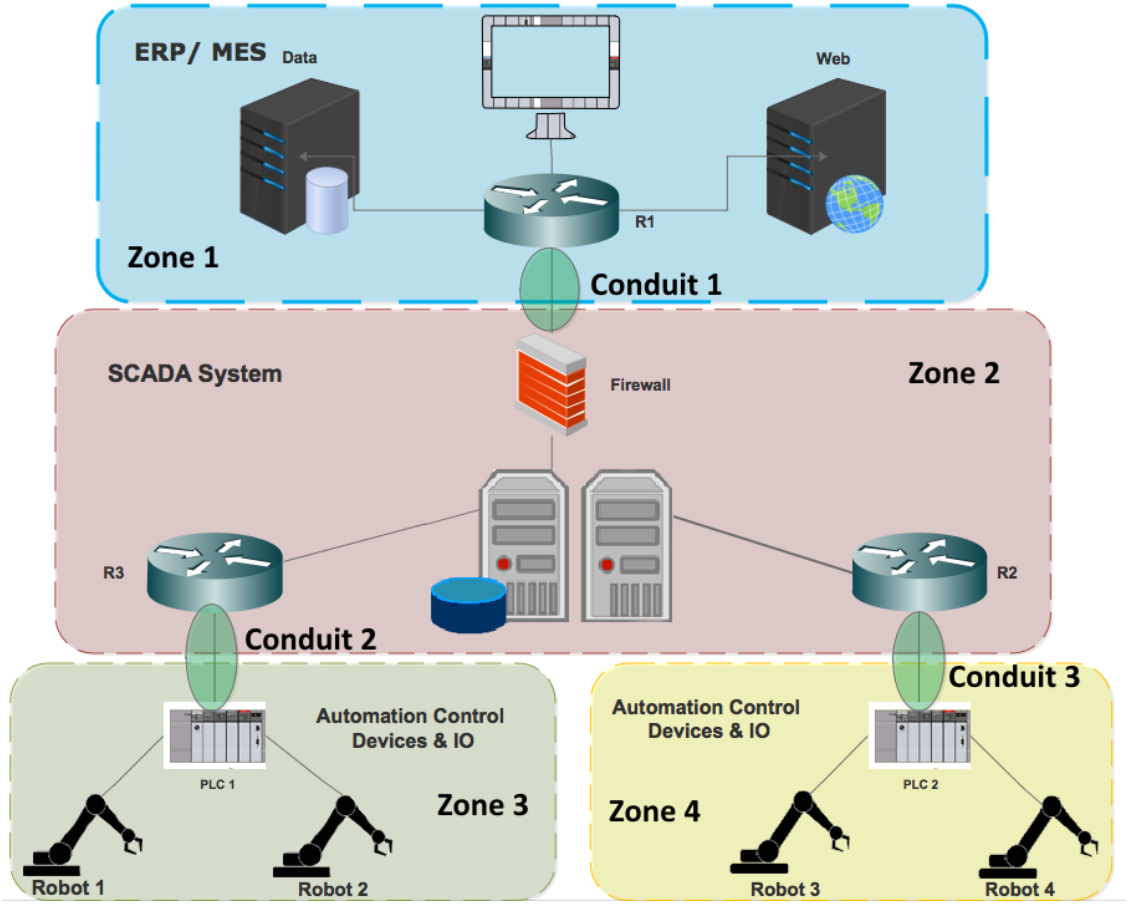


Figure 4.5: IEC 62443 zone and conduit concept

The break-down of the above example is performed in the security risk analysis phase. The interconnection and communication paths between the zones are then defined as “Conduits”, the pipes where secure data and information exchange is performed. The following section explains how this paper will use the concept of the zones and conduits not only for the overall operating plant but also for the single devices of the operating plant.

The Usage of IEC 62443

Nowadays, IEC 62443 is considered the most commonly used security standard in CPS and IoT application domains. In this research, we analyse security issues in the automotive domain for each vehicular element individually. As previously discussed, section 4-2 in the IEC 62443 standard series is focused on the components level. Furthermore, the IEC 62443-4-2 classifies requirements

into four component types, defined as follows:

- Software application requirements (SAR);
- Embedded device requirements (EDR);
- Host device requirements (HDR); and
- Network device requirements (NDR).

We classify automotive elements into these four groups, to facilitate the mapping process between security requirements and potential threats for a particular vehicular unit according to its type. In addition, the FRs can be adapted in this work to define a set of security objectives for addressing security issues.

According to the research gap **P3** (i.e., integrating relevant security requirements), we see that the IEC 62443 is the best set of relevant security requirements from relevant domains to be used and integrated within this research. These security requirements are used to prove the proposed cybersecurity framework's efficiency in handling and adapting relevant security requirements in the automotive domain.

4.1.2 Protection Profile

In order to build a security knowledge base for system engineering, it is essential to use an obvious approach to manage risk treatments and security standards. However, the only standards which define security requirements for a system are the IEC 62443-3-3 [isa13] and IEC 62443-3-1 [iec09]. Most standards focus instead on requirements for the security engineering process. This is done because system security requirements can change faster than standardization. Therefore, the requirements in IEC 62443 are difficult to use as input to define a knowledge base for security system engineering. Instead of looking at standards, we can consider open and certified documents based on standards. Furthermore, protection profiles are considered the best way to manage the security standards structure and describe the security considerations and resulting requirements for a Target of Evaluation (TOE) according to Common Criteria (CC). The protection profiles are based on common criteria ISO 15408 [ISO09b] and described in a structured way for generic types of system assets, risks, and controls to mitigate the risks. Figure 4.6 shows the mandatory structure of a protection profile, as described in [ISO09a].

The structure of the protection profile is described as follows:

1. **Introduction** summarizes the document and also gives a TOE overview description of the protection profile.
2. **Conformance Claim** discusses the conformance and relationships with other protection profiles.
3. **Security Problem** discusses the security problem definition includes assets, threats, and security assumptions, which will be discussed in Chapter 5.
4. **Security Objectives** defines a statement of the security objectives of TOE security. These objectives are intended to address security problems.
5. **Extended Components** defines extended requirements from common criteria.

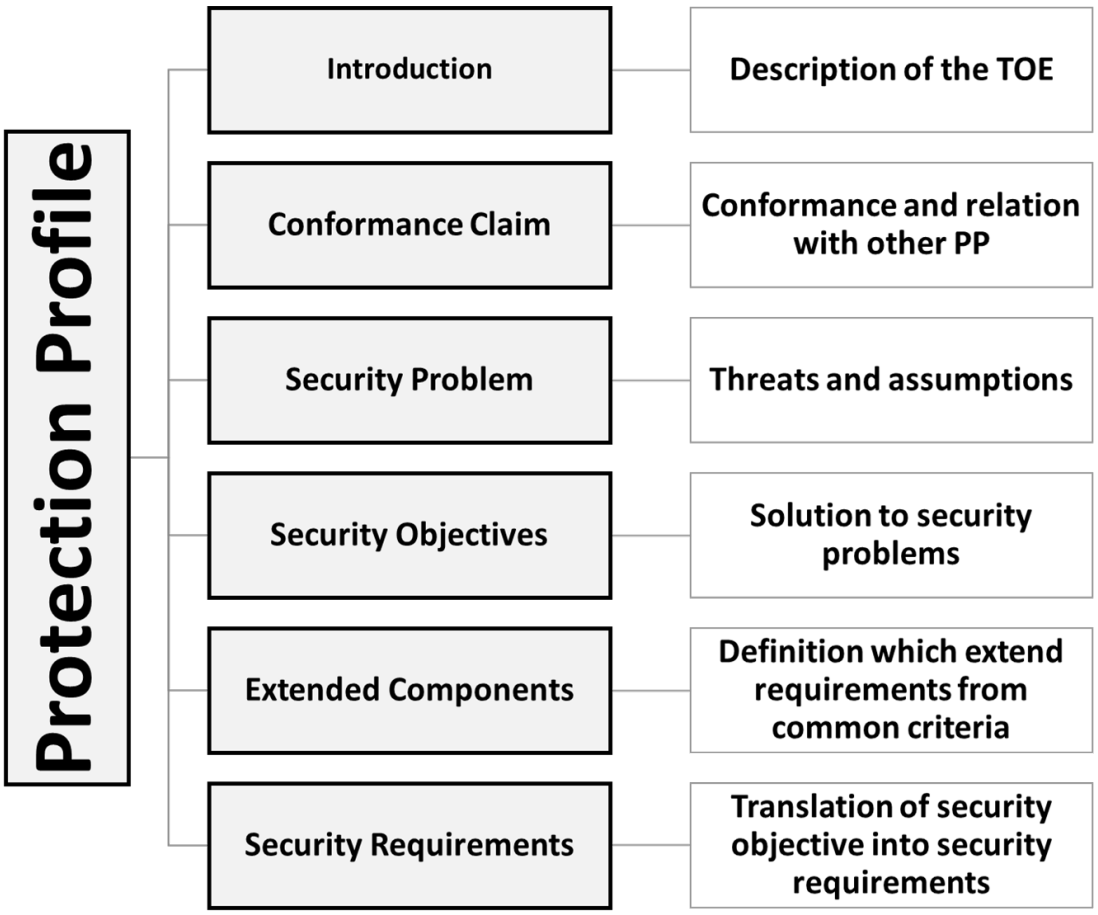


Figure 4.6: The structure of protection profile

6. **Security Requirement** translates the pre-described security objectives into a set of security requirements.

In this work, security requirements are defined in the V2X HSM protection profiles described in [Car18] [Car19]. These security requirements are used to address the identified potential threats and protect the assets of the specified TOE. While everybody can use protection profiles to define security requirements unambiguously, there are protection profiles certified by the governmental organization responsible for information and cybersecurity. For example, there are currently 202 protection profiles listed on the Common Criteria Port [2]. Each lists threats, security objectives, and security requirements for a generic type of system. Figure 4.7 illustrates the relationships between different elements in a protection profile.

The potential threats for a particular TOE will be discussed in Chapter 5.

4.2 Security Requirement Contents

The security requirements contexts are semantically analyzed to extract particular security properties. Common words from the data resources are used in this research to build the security

[2]https://www.commoncriteriaportal.org/pps/

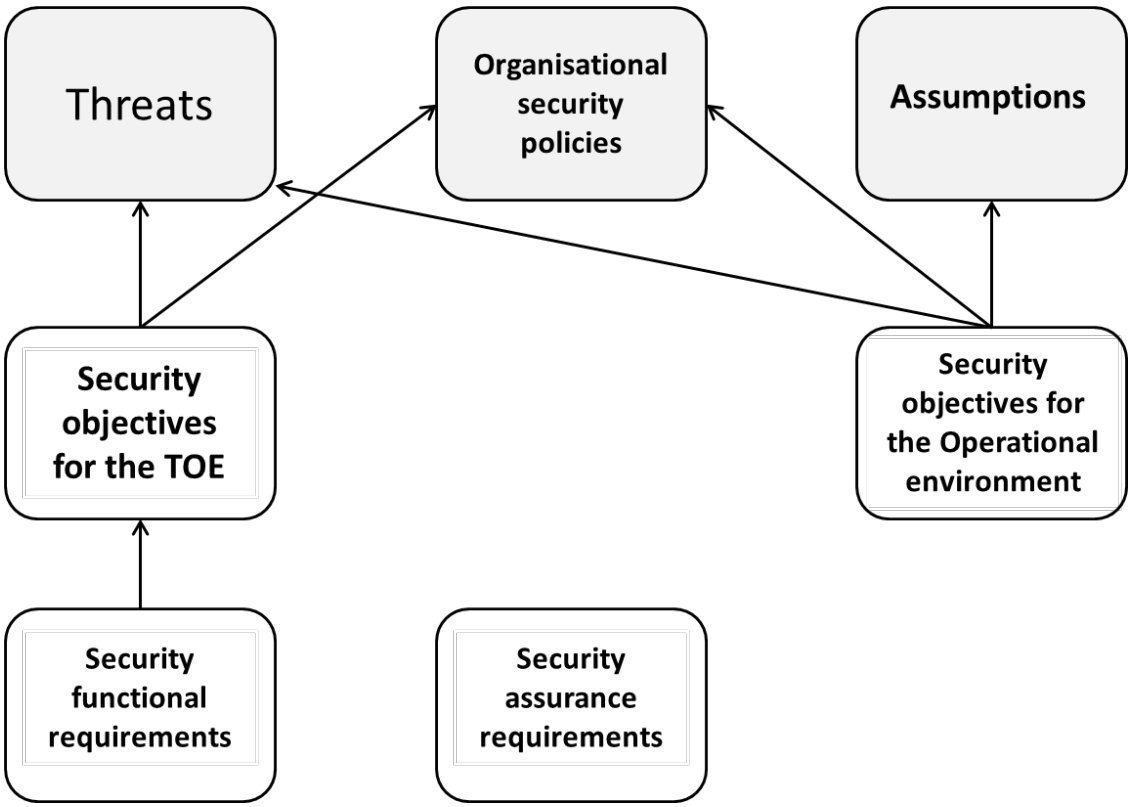


Figure 4.7: Relation of the elements in a protection profile [ISO09a]

requirements ontology structure. The data are extracted from a natural language text from the security standards IEC 62443 and security requirements of the protection profile (according to the common criteria). The data are defined in categories to define the ontology's main structure according to the type of the ontology hierarchy. According to the previously-discussed ontology structure in Chapter 3, we build this structure on spreadsheets to adapt the extracting data to be used in the ontology structure.

The security requirements shall be selected appropriately to address the potential threats and protect the vehicular components/assets from different forms of cyber-attack. Therefore, security requirement individuals shall contain the most helpful information for the selection and prioritization of the appropriate security requirements out of hundreds or thousands of other security requirements, in order to address threats and protect vehicular components/assets. The security requirements data are constructed in a spreadsheet with all relevant details from the sources to represent the original text's same content. The spreadsheet contains a set of details about the security requirements to reflect the same semantic of the security requirement content; these details are defined as follows:

ID: a unique code for the security requirements as described in the security requirement document:

- The IEC 62443-4-2 defines the security requirements for components, such as component requirements (CRs). These requirements are classified into four types according to the category of components (i.e., software application, embedded device, host device, and

network device). CR represents most of the document's requirements, but some specific components are classified according to the four types of components. These types of requirements are defined as follows [IEC19]:

  - Software Application Requirements (SAR).
  - Embedded Device Requirements (EDR).
  - Host Device Requirements (HDR).
  - Network Device Requirements (NDR).

- Therefore, another security property defined in the ontology structure is called "Type", representing the category of the security requirements according to the CRs.

- As in the protection profile, the security requirements are defined according to different categories based on the common criteria. Each category has a unique abbreviation that represents the category's content; therefore, these abbreviations are used in this work as an identification code for each security requirement.

Title: represents the title of the security requirement, as defined in the main resource document. The title also gives a short description of the security requirements' main functions, which could be a brief overview of the security requirement without going through the description section's details.

Properties: this field represents the security properties extracted from the main text of the security requirements defined in the main document sources. The properties are defined as data properties within the ontology context of each security requirement. Each security requirement establishes relationships with single or multiple security measures that share the security requirements' common security proprieties.

Security Levels As described earlier in this chapter, the security requirements in IEC 62443 are classified into four security-level capabilities (SL-C). The SL-C represents the security level in which the components of a system or the whole system can be achieved without extra measures [isa13]. These four levels are represented as follows [GGV15]:

- **SL1:** unintended,
- **SL2:** low resources, generic skills and low motivation (i.e., simple means),
- **SL3:** moderate resources, moderate motivation (i.e., moderate means),
- **SL4:** extended resources and high motivation (i.e., sophisticated means).

Security Requirements: represent the types of security requirements according to the seven foundational requirements (i.e., Identification and Authentication Control (IAC), Use Control (UC), System Integrity (SI), Data Confidentiality (DC), Restricted Data Flow (RDF), Timely Response to Events (TRE), and Resource Availability (RA).

Dependencies: some security requirements have a sub-set of security requirements which are considered a subcategory of the main security requirement based on the primary CR. Therefore, this data property is used to define the dependency of the security requirement.

4.3 Security Requirements: Ontology Building

This section displays the next step in this research context, which aims to convert all previously extracted data into the ontological form. The previously discussed contents of security requirements are defined in a spreadsheet. The ontology structure will then be used to apply a set of operations for performing the security verification validations of the previously selected security requirements. The ontology is also used to check the protection level of components/assets in a vehicular network. In this work, we use Cellfire [3], as a plugin for the Protégé as the most well-known ontology editor. Cellfire is used for importing the data in the spreadsheet into an ontological representation form [joh18]. For this purpose, we need to create a set of expressions for mapping from the spreadsheet into the ontology structure. We use a domain-specific language (DSL) to create these expressions, as discussed in [O'C20].

These expressions recognize the content of columns and rows in the spreadsheet to create the required ontology structure. For example, security requirements contain a set of details (i.e., ID, name, security properties, and others) described as a set of data properties that represent the individual's contents. Figure 4.8 illustrates a set of expressions written to select a specific set of data from particular locations (i.e., columns) to import and convert the tabular data representation into an ontological representation.

The figure shows a set of expressions used to import the spreadsheet's security requirements from row number "5" until the end (i.e., the end of lines command is represented by a '+' sign). The importing process includes all columns starting from column "C" until column "AA". Expressions describe the data property and object property for each individual (i.e., in the security requirement) to create an ontology entity containing all relevant details, representing the primary resource's main content. After we have applied these expressions, new axioms will be created and integrated with the ontology structure. These expressions can also define the object property between the individuals in the ontology structure to define relationships between entities. For example, as discussed before, security requirements have a set of security measures; the "hasProperty" relationship is identified to represent that a security requirement has single or multiple security measure(s). Security measures could also be used to define the security mitigation of a particular vehicular component/asset. Therefore, the "hasMitigation" can also be expressed to define the relationship between vehicular components/assets with relevant security measures.

4.4 Chapter Summary

This chapter discussed data sources of security requirements that are used in this research. The data are collected from different resources used to build a comprehensive security requirements ontology structure and integrated with this research context. A particular set of data properties is extracted from the data resources; then, we build a tabular form of these extracted data to aggregate all the contents from resources to define the ontology hierarchy's main context. The Cellfire plugin

[3] https://github.com/protegeproject/cellfie-plugin

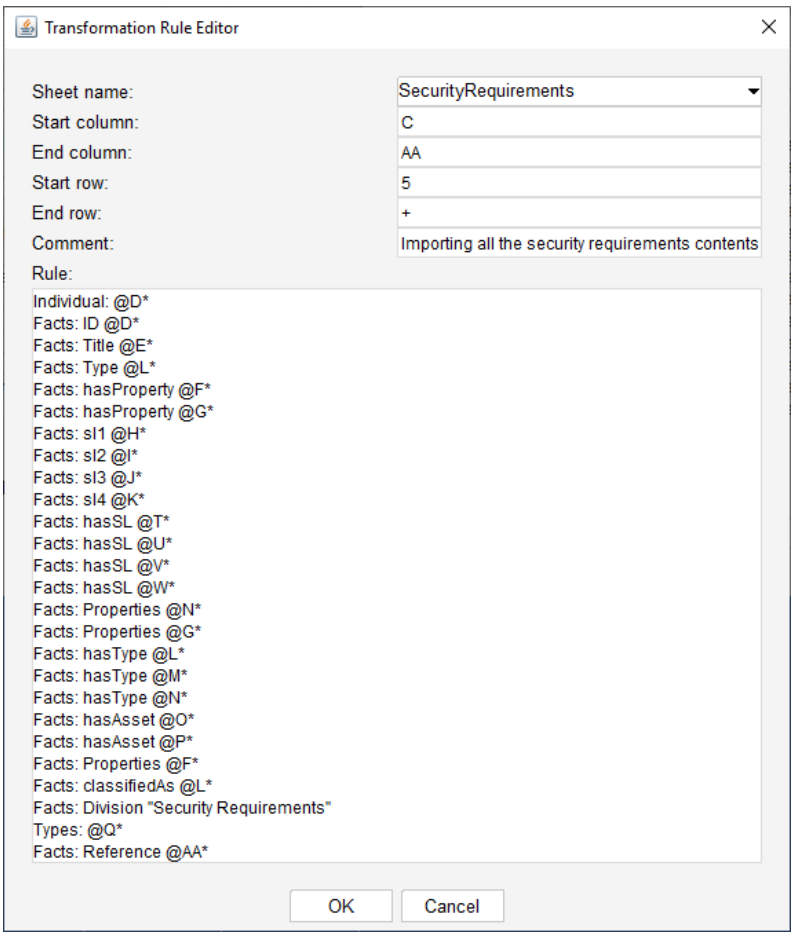


Figure 4.8: A set of created expressions for importing security requirements data from the spreadsheet into the ontological form

for Protegé ontology editor is used to import the context from the spreadsheet into an ontological representation.

The next chapter discusses the data resources used to collect the relevant potential threats in the vehicular domain. There are multiple resources selected and integrated within this study; all of these resources will be covered and discussed in Chapter 5.

5 Research Data Collection: Potential Threats and Vehicular Components/Assets

This chapter discusses potential threats, security vulnerabilities, and vehicular components/assets used in this work. Studying potential threats in this study is essential for identifying security concerns in the automotive domain to determine the specific security requirements that resolve these security issues, which is considered one of the key research objectives. The chapter is divided into three main sections; section one includes a discussion on the sources of potential threats. The second part involves the collected data's adaptation process to extract the original text's main common properties to construct a tabular representation of threats data. The last section presents a method to convert the tabular representation form into the hierarchical ontology representation.

This chapter includes content from an article published within this thesis work. The following refers to the selected publication that is integrated within the context of the chapter:

- *Schmittner, Christoph, Abdelkader Magdy Shaaban, and Johannes Hellrich. 2019. "Automating the Construction of a Security Threat and Mitigation Pattern Library." in INCOSE Artificial Intelligence for Systems Engineering.*

5.1 Data Sources

Potential threats and vehicular components/assets selected in this work are based on multiple data sources. It is essential to choose reliable data with a high level of trust in order to validate the work's effectiveness. Therefore, in this research we use the UNECE threats list [UNE17b], V2X Hardware Security Module Protection Profile [Car19], [Car18], and Common Vulnerabilities and Exposures (CVE) [Com20].

5.1.1 The UNECE Threats List

As discussed in the "UN Regulation on uniform provisions concerning the approval of vehicles about cybersecurity and their cybersecurity management systems," [TC20] the approval authority or the technical service shall verify that the vehicle manufacturer has a cybersecurity management system and verify its compliance with the regulation. As mentioned in the regulation, the approval authority or technical service should check that the vehicle manufacturers apply their cybersecurity management system through the different phases of the vehicle cycle, such as development, production, and post-production phases. The vehicle manufacturer shall also demonstrate cybersecurity management system is adequality considered, including risks and mitigations presented in Annex 5 in [TC20]. The Annex consists of three main sections; Section A contains a list of threats, vulnerabilities, and attack methods. Section B discusses a set of security mitigation (i.e., set of measures for reducing a risk) for threats within a vehicle, where Section C, includes security mitigations relevant to threats outside the vehicles.

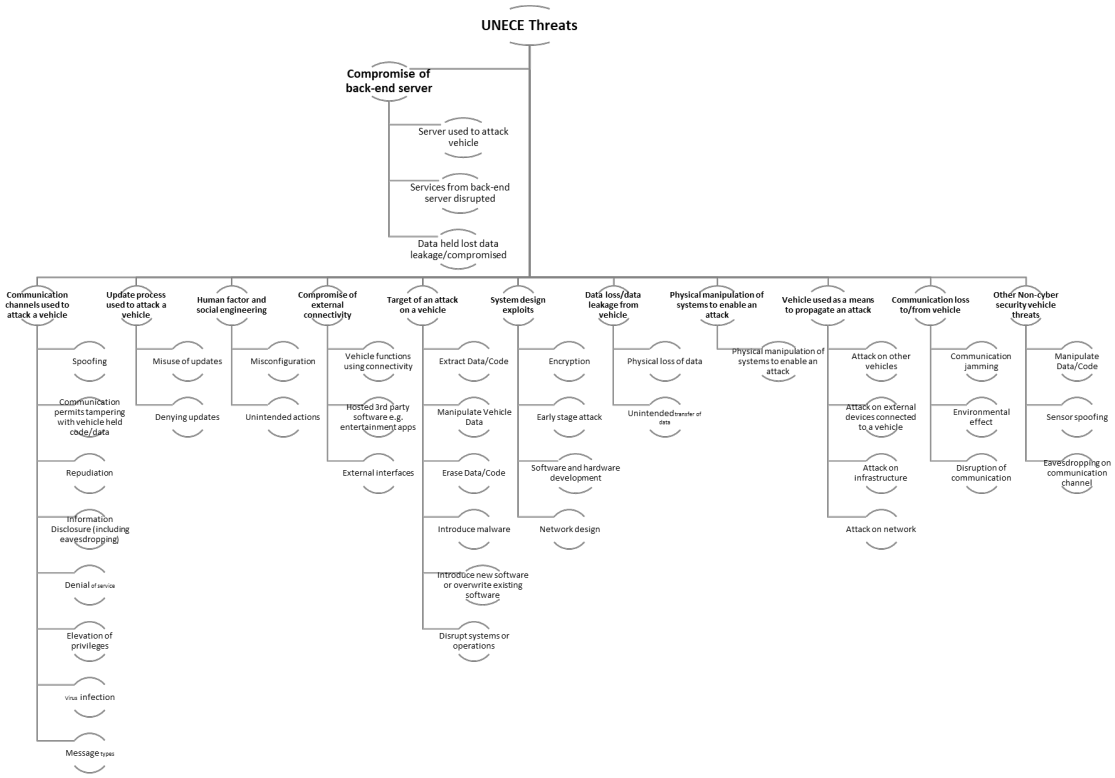


Figure 5.1: The structure of the UNECE's threats list

One of the leading research objectives in this work is introducing a cybersecurity framework conducted with realistic case-studies in the automotive domain. Therefore, this list of threats and security mitigations are integrated into this work to build a complete ontology structure combined with alternative automotive case-studies. We used another version of this threats list created in a spreadsheet defined by the UNECE that can be found on [UNE17b], which is used in this work to provide more details about threats. Figure 5.1 depicts the structure of the threats and their categories according to the UNECE's threats list.

The spreadsheet has more details about threats; that is why this version of the threats list is integrated with this work to give more in-depth details about threats for building a comprehensive ontology hierarchy model in the automotive domain containing all information on threats. This information is useful for defining more relevant details about threats; the classification of this information describes as follows:

Attack/Vulnerability: the classification of the threat could be an attack or an existing security vulnerability in the vehicular system.

Type of entry: describes the attack entry point into the vehicular network either by cyber or non-cyber methods.

Threat effect: describes the effect of a threat against the vehicular network, which could be a direct, cascading, or vulnerability outcome.

Access Method: defines some details about the access methods into the vehicular network:

- **Threat path:** defines a short description of the threat path, which could be multiple hops or single hop.
- **Physical attack:** indicates that physical methods could be used to trigger a particular threat.
- **Remote attack:** indicates that remote methods could be used to trigger a particular threat.

Some other details are defined in the spreadsheet, representing the vehicle architecture components where the attack may be initiated. This information is classified into four main categories, as follows:

- Internal architecture,
- Physical,
- Wireless, and
- External.

In this research, we extend these proposed values to calculate the probability of threats that could be triggered or propagated within the vehicular network's internal/external structure. The results of these evaluations are considered as a set of likelihood parameters. More details about this evaluation process are discussed in Chapter 6. The last part of this spreadsheet of potential threats is the potential impact, which describes the impact of each threat, classified into seven categories, as follows:

- Safe operations of vehicle affected,
- Vehicle functions stopped working,
- Software modified, performance altered,
- Software altered but no operations effects,
- Data integrity breach,
- Data confidentiality breach, and
- Other, including criminality.

These seven categories are updated and used in this research to evaluate the impact level of each threat on this sheet separately. Estimating the likelihood and impact parameter values is used in the risk evaluation process of each threat's overall risk to estimate the potential severity level. This estimation process is discussed in Chapter 6.

Example of Potential Threats

As discussed earlier, the UNECE threats list combines a wide range of potential threats that have a negative impact on the automotive domain. Threats are arranged into multiple groups which classify the target element that an attacker follows in order to reach a malicious objective. Threats could be initiated within the vehicular network, such as abuse of privileges of vehicular components, breach the confidentiality and integrity of the vehicular data, vehicular physical manipulation, and others. In addition, some security weaknesses could be exploited through different external attack methods to affect the vehicle operation, such as an attack on the OEM's server, compromising the external connectivity of vehicles, and others. An attacker could also control a vehicle to propagate a set of adverse events against the other connected cars. Figure 5.2 illustrates a conceptual model of the internal components in a vehicle and depicts the connectivity with the external interactions (e.g., vehicles, OEM's back-end servers, infrastructure, and so on). This is an updated and extended version of "CAV Ecosystem figure", as depicted in [UNE17a].

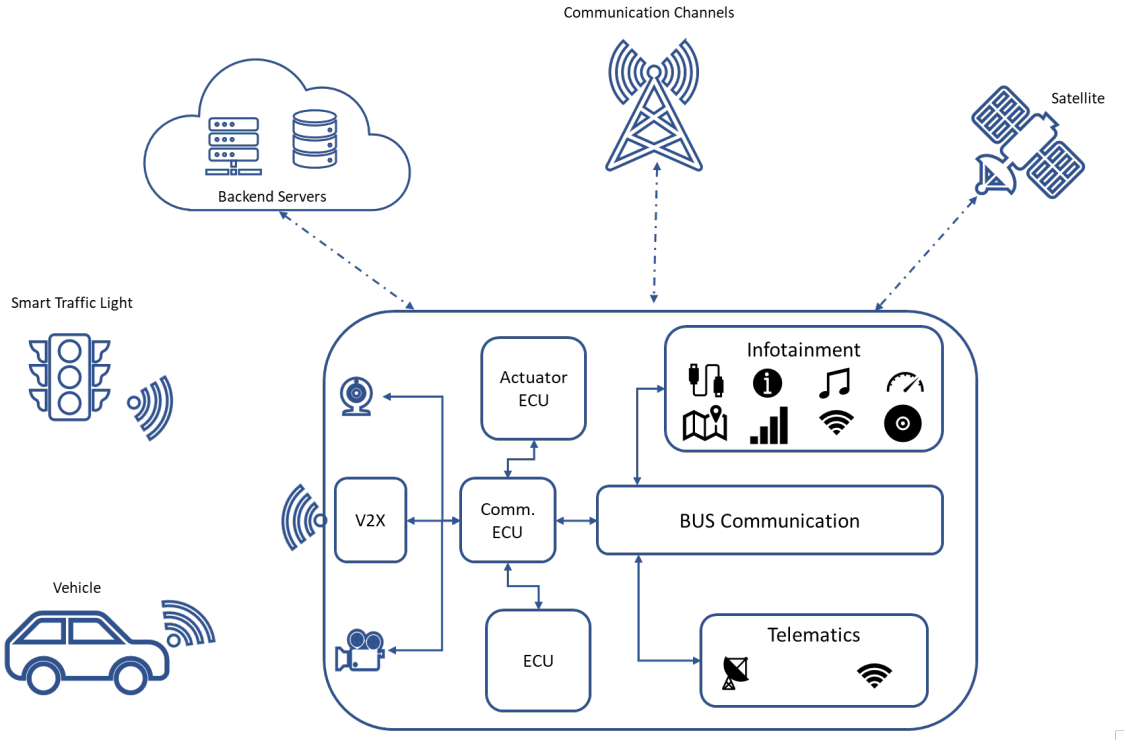


Figure 5.2: Example of connected and autonomous vehicle

As illustrated in the figure, all the vehicle's internal components are connected through a bus communication channel. The vehicle contains electronic/electrical units such as electronic control units ECUs and communication ECUs. An infotainment unit is connected to the other units, through the main communication bus. The infotainment is integrated into this model, as a vehicular unit provides different services that could be considered vulnerable points by attackers who might use them for malicious activity. A USB port could be utilised to install malicious software. Telematics is also included, which provides connectivity with external entities such as cloud and cellular communication. A V2X device provides the vehicle with communication with other external nodes such as vehicles, infrastructure, and other smart units.

The UNECE threats list helps define potential threats that could negatively impact different components described in Figure 5.2. For example, there is a set of threats identified in the list relevant to the back-end server. This group contains a set of threats that have a direct negative impact against servers, as follows:

- Attack on the back-end server stops it from functioning.
- Loss of the information in the cloud.
- Unauthorised internet access to the server.
- Unauthorised internet access to the server.

This is an excerpt of some selected threats that negatively impact the backend sever. Even though these kinds of attacks could have no direct impact on the vehicle, these threats could negatively affect the vehicle, such as sending falsifying data or disclosing personal information.

5.1.2 CVE - Common Vulnerabilities and Exposures

Another threat resource considered in this research is the Common Vulnerabilities and Exposures (CVE) [1]. According to Ref. [Com20], the CVE is considered a security vulnerability and exposure reference, which could be integrated with products and services according to the official terms of use. The case study used in this work contains a Linux operating system as a part of the example. The previously discussed threats do not contain threats related to the Linux operating system. Therefore, we use the CVE in this work to select a small set of vulnerabilities relevant to the Linux operating system, which helps to define a realistic set of potential threats in our use case study, as described and discussed in Chapter 7.

The CVE is used by organizations to cover a wide range of multiple vulnerabilities and assist in enhancing security [GW09]. Each published vulnerability has a collection of information, providing more detail about the detected vulnerability. Figure 5.3 shows a simple example as a screenshot for a Linux operating system from the CVE website [CVE18].

Vulnerability descriptions consist of various information supporting a detailed explanation of a specific vulnerability.

- **CVSS Scores & Vulnerability Types:** As discussed before in Section 2.2.1, the CVSS applies to define the severity level of the IT vulnerabilities. Therefore, the CVSS is applied here to estimate the severity degree for each identified vulnerability. Hence, this section includes a set of metrics (i.e., Basic Metric to provides more security information regarding this vulnerability, as discussed 2.2.1). According to [MSR07] these information are defined, as follows:
  - **Confidentiality Impact:** Represents the impact level on confidentiality, in case of the vulnerability is exploited successfully.
  - **Integrity Impact:** Describes the integrity impact level when a vulnerability is exploited.
  - **Availability Impact:** Same as the confidentiality and integrity, is a metric level representing the level of impact on availability.

[1] https://cve.mitre.org/index.html

Vulnerability Details : CVE-2008-4306

Buffer overflow in enscript before 1.6.4 has unknown impact and attack vectors, possibly related to the font escape sequence.
Publish Date : 2008-11-04 Last Update Date : 2018-10-11

Collapse All Expand All Select Select&Copy Scroll To Comments External Links
Search Twitter Search YouTube Search Google

– CVSS Scores & Vulnerability Types

CVSS Score	9.3
Confidentiality Impact	Complete (There is total information disclosure, resulting in all system files being revealed.)
Integrity Impact	Complete (There is a total compromise of system integrity. There is a complete loss of system protection, resulting in the entire system being compromised.)
Availability Impact	Complete (There is a total shutdown of the affected resource. The attacker can render the resource completely unavailable.)
Access Complexity	Medium (The access conditions are somewhat specialized. Some preconditions must be satistified to exploit)
Authentication	Not required (Authentication is not required to exploit the vulnerability.)
Gained Access	Admin
Vulnerability Type(s)	Overflow
CWE ID	119

– Additional Vendor Supplied Data

Vendor	Impact	CVSS Score	CVSS Vector	Report Date	Publish Date
Redhat	moderate	4.4	AV:L/AC:M/Au:N/C:P/I:P/A:P	2008-10-28	2008-10-29

If you are a vendor and you have additional data which can be automatically imported into our database, please contact admin @ cvedetails.com

Figure 5.3: Vulnerability details of CVE's example, this example is a screenshot from the CVE-2008-4306, as described in [CVE18]

- **Access Complexity:** Represents the level of complexity of an attack that necessitates exploiting a particular vulnerability.
- **Authentication:** Specified whether authentication is required to exploit a vulnerability.
- **Vulnerability Type(s):** Defines the category of vulnerability such as overflow, denial of service, execute code, and others.

Other details are included to provide more information that focuses on product vendor and version affected by security vulnerability, such as:

- Additional Vendor Supplied Data.
- Related OVAL Definitions.
- Products Affected By CVE-****-****.
- Number Of Affected Versions By Product.
- References For CVE-****-****.

5.1.3 Protection Profile

As discussed in Chapter 4, the protection profile is considered to be the best way to manage the structure of security requirements and describe security considerations and resulting requirements for a Target of Evaluation (TOE) according to Common Criteria (CC). The protection profiles are based on common criteria ISO 15408 [ISO09b] and described in a structured way for a generic type of system assets, risks, and controls to mitigate the risks. The integrated protection profiles in this work [Car18] [Car19] consist of a set of potential threats that threading the TOE (i.e., V2X HSM). The protection profile discusses the security objective and security requirement for

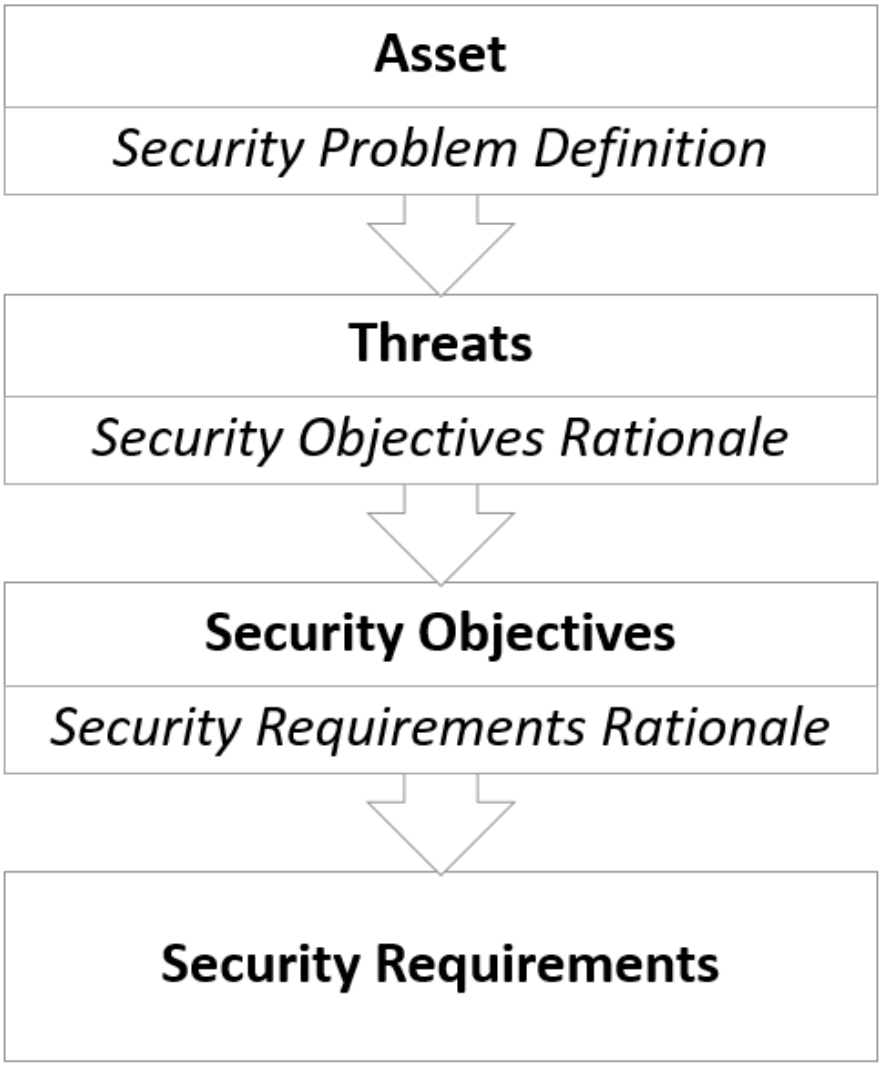


Figure 5.4: Example of a protection profile and the defined relations

protecting user data against external access and modification, as depicted in Figure 5.4. This is defined in the Protection Profile for Trusted Platform for secure communications [Cen16].

The PP [Cen16] describes the TOE as a mobile device that provides trusted mechanisms for secure communications with other devices. For example, it can be used for communication and voice applications using a trusted channel. The channel is a Virtual Private Network - VPN - to provide confidentiality, integrity, and authentication. The TOE contains different assets that need to be protected by the TOE [Cen16]. The following is a list of some selected assets related to the TOE:

- **AS.USE_DATA:** user-related data such as contacts, SMS, call registers, and so on.
- **AS.VPN_MODULES:** software and modules are used to establish VPN connections.
- AS.INSTALLED_APPLICATIONS: authorized applications that are installed to be in the TOE.

Table 5.1: Security objectives and threats related to the AS.USE_DATA asset

		Threats					
		T.USR_DATA	T.UNAUTH_BOOT	T.BYPASS	T.UNAUTH_VPN:	T.ATTACK_VPN	T.UNAUTH_COM
Security Objectives	O.ERASURE	X					
	O.AUTHENTICATION	X		X			X
	O.SECURE_BOOT		X	X			X
	O.TUNNEL			X	X	X	X
	O.INTEGRITY				X	X	X
	O.SELFTEST				X	X	X
	O.SECURITY_POLICIES				X		
	OE.KEYS					X	

- **AS.OS:** operative system and core software, which runs the installed applications.
- **AS.CRYPTOGRAPHIC_ASSETS:** cryptographic material stored and handled inside the TOE.

Then there is a list of threats to define the potential security problems, according to the AS.USE_DATA asset that affects this asset (i.e., AS.USER_DATA,) and other ones. As defined in [Cen16], here is a list of threats:

- **T.USR_DATA:** the threat applies to all external TOE interfaces.
- **T.UNAUTH_BOOT:** bypass the encryption to be able to boot and start up the TOE.
- **T.BYPASS:** bypassing the authentication mechanism of the TOE to unlock its feature.
- **T.UNAUTH_VPN:** extract confidential communications and bypass the security mechanisms established to force the TOE to communicate through this channel.
- **T.ATTACK_VPN:** disclose information that is communicated between the TEO and other points.
- **T.UNAUTH_COM:** establish an unauthorized communication channel using available interfaces of the TOE.

The security objectives are then defined to provide precise solutions to security problems, according to the threats mentioned above related to the AS.USE_DATA asset, Table 5.1 gives a summary of all security objectives related to these threats, according to to [Cen16].

Eight security objectives are selected according to the previously defined threats as described in [Cen16]; these objectives are defined as follows:

- **O.ERASURE:** The TOE shall perform secure removal of personal data according to requests from an authorized user.
- **O.AUTHENTICATION:** TOE shall authenticate the authorized users to control the access into the user data.

- **O.SECURE_BOOT:** This addresses the secure boot and the operation chain processes.
- **O.TUNNEL:** Gives trusted channels that will control the traffic flow with outside units.
- **O.INTEGRITY:** Verifies the integrity of cryptographic mechanisms and VPN modules.
- **O.SELFTEST:** The TOE shall be able to perform self sets during the initial start-up to prove the correct operation.
- **O.SECURITY_POLICIES:** The TOE shall add a set of policies to prevent unauthorized access into security features are managed by the TEO.
- **OE.KEYS:** keys are used for encrypting TOE data storage.

A set of security requirements is then defined according to the discussed security objectives to address security problems. Table 5.2 gives an overview of the selected security requirements for the asset AS.USE_DATA asset as presented in [Cen16].

Table 5.2: Security objectives and security requirements of the AS.USE_DATA asset

		Security Objectives					
		O.ERASURE	***O.AUTHENTICATION***	***O.SECURE_BOOT***	***O.TUNNEL***	***O.INTEGRITY***	***O.SELFTEST***
Security Requirements	***FDP_ZER.1***	X					
	FIA_AFL.1	X	X				
	FMT_SMF.1	X	X				
	FPT_TST.2	X		X		X	X
	FDP_DSK.1		X	X			
	FIA_UAU.2/KEK		X	X			
	FIA_UAU.2/PIN		X	X			
	FIA_UAU.2/KEY-admin		X	X			
	FPT_SBT.1			X			
	FTP_ITC.1/CIK-tunnel			X	X		
	FDP_IFC.2				X		
	FDP_IFF.1				X		
	FMT_MSA.1				X		
	FMT_MSA.3				X		
	FPT_STM.1				X		
	FTP_ITC.1/VPN-tunnel				X		
	FTP_ITC.1/REM-ADM				X		
	FTP_ITC.1/AUDIT				X		
	FPT_FLS.1					X	X

This example shows only an excerpt of security objectives and requirements. Such information can be used to define a potential security pattern to address user data protection against external access. There are still some open issues. Despite the general format of the protection profile being defined, there is some freedom in how the template is used and the level of information detail. This means that it can be difficult to process protection profiles.

5.1.4 Vehicular Components/Assets

The key research goal is to secure vehicle components/assets from cyberattacks; these components/assets are the backbones of this study, as explained in Chapter 7. This research derives the related vehicle components/assets from the selected potential threats and defines the relevant security mechanisms to protect these components/assets from any malicious actions. It also describes a wide range of vehicle components, such as:

- V2X unit.
- Electronic Control Unit (ECU).
- Human-Machine Interface.
- communication Interfaces.
- Operating System (e.g., Linux).
- Wire Interfaces.
- Wireless Interfaces.
- Multiple assets related to the TOE as specified in [Car18] [Car19].

5.2 Potential Threats Contents

Potential threats description is studied and analyzed to derive particular security characteristics. Common properties from the data resources are used in this research to build the complete set of potential threats in an ontology structure. The data are extracted from a natural language text from the UNECE threats list, CVE description for particular vulnerabilities, and potential TOE threats in the protection profile. The data are then described in categories and sub-categories to represent the main structure of the threats ontology model. According to the previously discussed ontology structure in Chapter 3, we build this structure on spreadsheets to adapt the extracting data to be used in the ontology structure.

The spreadsheet is used here to define the threats' content in a tabular manner to define all collected contents about the threats' malicious behaviour before converting this data form into an ontology representation. The form of the threat's spreadsheet has a different structure than the security requirements. The content of the spreadsheet is described as follows:

ID: special identification code of each threat, which represents a primary key for each threat to differentiate between various ones.

Category of Threats/ subcategory: as described in Figure 5.1, each threat is outlined under the main categories of threats. Each category has at least one subcategory. Therefore, the spreadsheet contains two columns to imitate the main structure of the original document's threats.

Title: represents the name of the threats, as defined in the leading threat document. The title also gives a short description of the threat of its malicious behaviour, which could be a brief overview of the threat without going through the description.

Description: a description field is specified in this work to contain all description content of each threat, as described in the main document. Based on each threat's description content, the affected vehicular units (i.e., components/assets) are identified. Also, the relevant security properties that could be breached by threats are extracted to understand each component/asset's principal security vulnerabilities. These vulnerabilities play an essential role in selecting the applicable security requirements to address these threats and protect the vehicular components/assets.

Risk Evaluation: is listed in the spreadsheet in order to reflect risk estimation values for each threat. This evaluation is based on a standardized assessment approach to measure each threat's entire risk and identify the critical severity level. The risk assessment section in the spreadsheet contains four main parts, as follows;

- **Likelihood:** the evaluation of the likelihood parameter values will be discussed in Chapter 6. The values are evaluated according to access methods that are defined in the UNECE threats list as discussed earlier in Section 5.1.1
- **Impact:** the same as the likelihood parameter values, these values will be discussed later in Chapter 6. However, as presented in the UNECE threats list, potential impacts are used to evaluate the parameter values of the impact level of each threat.
- **Risk:** the overall risk is estimated for each threat in order to identify the risk level. The risk level helps in prioritizing the risk for each threat and identifying the severity level.
- **Risk Severity:** is identified based on estimated risk assessment values. In this work, four key risk severity levels are followed to define each threat's main actual severity level. These levels play an essential role in selecting security requirements to meet the main security goal. For example, the SL-4 should at least address a threat as critical severity. However, the high severity can be addressed by either SL-3 or SL-4. The severity levels are assigned as follows:
  - low risk,
  - Medium risk,
  - High risk, or
  - Critical risk.

5.3 Components/Assets Contents

In this research, the components/assets play an important role as they are considered the target that attackers need in order to achieve their malicious purpose through various potential threats.

On the other hand, they are essential in selecting security requirements to protect the vehicle from various attack scenarios. The selected components/assets are specified in a tabular representation in the spreadsheet to integrate all components/assets with relevant information to facilitate the converting process into the ontological representation. Each component/asset has a collection of security measures used to protect against various attack scenarios. In this work, a set of different security measures is integrated with components/assets; some of the selected security measures are defined as follows:

- **Access Control:** a protection mechanism against unauthorized access into the system's units.
- **Authentication:** aims to verify a unit's identity in the system; this unit could be human, software, hardware, or process.
- **Authorization:** checks the authorization privilege of a particular process or user.
- **Confidentiality:** aims to keep the system data confidential to keep it defended from any non-disclosure attacks.
- **Integrity:** aims to check the correctness of the data content.
- **Input Validation:** checks the incoming data's validation to ensure that the received content is out of any malicious code.
- **Secure Storage:** aims to secure the storage of the system/system's unit data.
- **Reliability:** checks the trustworthiness of the communication channels.
- **Latency:** aims to check the time elapsed between the data transmission until it reaches the receiver point.
- **Tamper Protection:** intends to protect the system configuration or data from unauthorized manipulation.
- **Secure Boot:** aims to verify the trustworthiness of firmware's running code.
- **Backup:** attempts to copy the critical system data into another safe storage place to avoid data damage or loss.
- **Updates:** provides the ability of the system's software to be up-to-date for fixing existing bugs or adding additional features.
- **Cryptograpy:** aims to encrypt/decrypt all of the data to keep it confidential to avoid data disclosure or eavesdropping attacks.
- **Administration:** represents the management process of software or hardware in a system.

Each component/asset has other information described in the spreadsheet to give more details on each vehicular unit. As discussed before, the security requirements are classified into four main categories (i.e., Software Application Requirements (SAR), Embedded Device Requirements (EDR), Host Device Requirements (HDR), and Network Device Requirements (NDR)) according to the CRs. Therefore, the components are classified into two main divisions. These divisions are defined as follows:

Category and Superclass: these fields represent the main category and class for each component/asset according to its feature and functionality. For example, the vehicle interacts with the OEM server to share information and keep the car updated. Therefore, we described this unit's main category as "OEM Backend" and classified the same unit's superclass as a host device.

5.4 Potential Threats: Ontology Building

As discussed in 4.3, we use the same way to define a similar set of expressions for mapping from the spreadsheet into the ontology structure. These expressions are written to recognize the content of the columns and rows within the spreadsheet to create the required ontology structure of the potential threats. Figure 5.5 illustrates a set of expressions written to select the specific threats data from particular locations (i.e., columns) to import and convert the tabular data representation into an ontological representation.

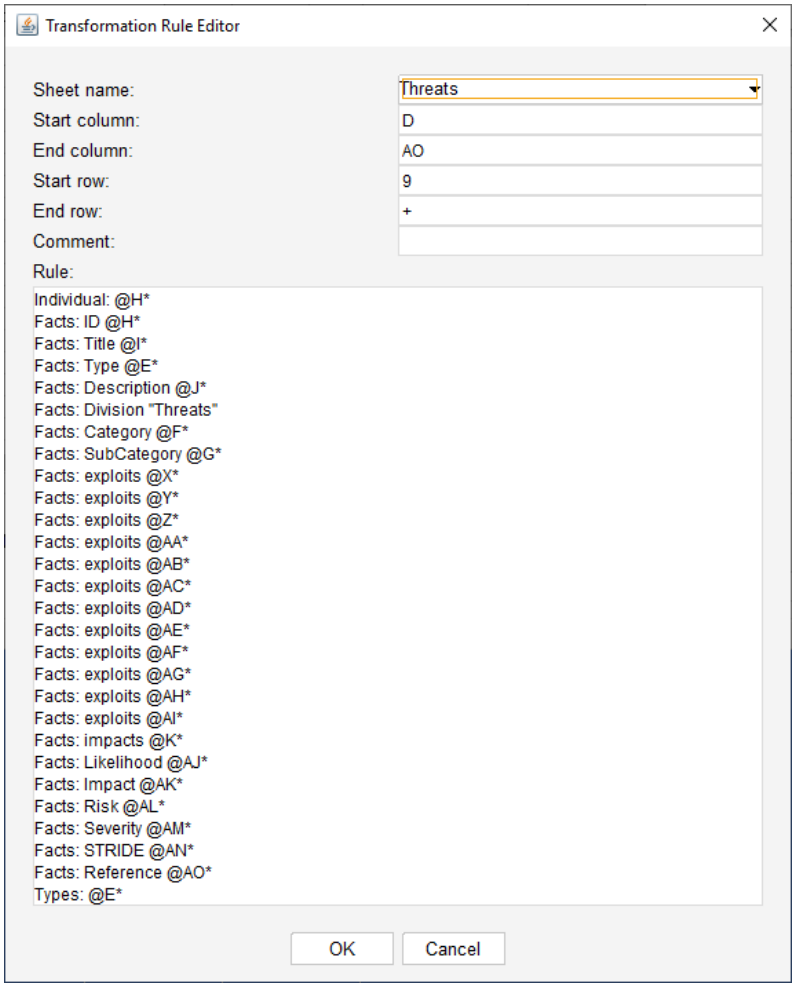


Figure 5.5: A set of created expressions for importing threats data from the spreadsheet into the ontological form

The expressions describe the data property and object property for each individual (i.e., in the potential threats) to create an ontology entity containing all relevant details, representing the

primary resource's main content. After we applied these expressions, new axioms will be created and integrated with the ontology structure. These expressions can also define the object property between the individuals in the ontology structure to define the relationships between the entities. For example, the "exploits" relationship is identified to represent the relationship between a threat and the security properties. The main use-case used in this research is built in the same way by defining the relationships between the different entities in the ontology structure. The case-study and its contents will be discussed in Chapter 7.

5.5 Chapter Summary

This chapter discussed the types of potential threat data resources that are used in this research. The data are collected to build a comprehensive threats ontology structure in the vehicular domain to be integrated with this research context. Relevant vehicular components/assets are derived from the potential threats resources to define the common vehicular units that fit this research context. A set of expressions is written and managed by Cellfire to import the threats and vehicular components/assets contexts from spreadsheets into an ontological representation.

The next chapter discusses the proposed cybersecurity framework's primary structure and defines the building blocks of our proposed approach for verifying and validating the selected security requirements. Also, it will explain how the proposed framework can select a new set of security requirements integrated within the ontology hierarchy to protect the vehicular components/assets from different cyber attacks while addressing the existing potential threats.

6 An Ontology-Based Cybersecurity Framework for the Automotive Domain

Chapter 4 and Chapter 5 discuss how threats and security requirements from different resources can be developed and adapted in this study. This chapter highlights the fundamental structure of the proposed cybersecurity framework. It begins with describing how the proposed framework perceives and recognizes the contents of the ontology and the relationships between its entities, it then provides a discussion on the various actions applied to the ontology model to estimate the correctness of the selected security requirements against potential existing threats.

This chapter includes content from some articles published within this thesis work. The following list refers to the selected publications that are integrated within the context of the chapter:

- *Shaaban, Abdelkader Magdy, Christoph Schmittner, Thomas Gruber, A. Baith Mohamed, Gerald Quirchmayr, and Erich Schikuta. Ontology-Based Security Requirements Framework for Current and Future Vehicles. Vol. Data Science and Big Data Analytics in Smart Environments. CRC Press Taylor & Francis Group. - (Accepted)*

- *Shaaban, Abdelkader Magdy, Christoph Schmittner, Thomas Gruber, A. Baith Mohamed, Gerald Quirchmayr, And Erich Schikuta. "Automated Ontology-Based Security Requirements Identification for The Vehicular Domain." Journal of Data Intelligence 1, No. 4 (2020): 401-418.*

6.1 Conceptual Architecture of the Framework

This section discusses the structure of the proposed ontology cybersecurity framework. The framework is developed to be fully-adaptable to handle different types of input to perform its action on the vehicular model. Chapter 7 discusses more details on the different types of inputs the proposed framework could accept for investigating the applied security requirements and filling the existing security gaps with the applicable set of security requirements. The framework evaluates the security level that is achieved after the security requirements are applied to the vehicle. Then it provides a new range of security requirements that could be applied to the vehicle to improve that level and meet the actual security goal. Security procedures are defined as a series of activities that use objects aimed at preparing, planning, defining, executing, and reviewing test results and revisiting preparation [SGS12]. Automotive security testing requires a robustness method to manage all vehicle security details and manage a wide range of security requirements to ensure a vehicle's security level. Therefore, the proposed model aims to automate the validation process of security requirements based on testbed execution. Figure 6.1 illustrates the metamodel of the proposed ontology-based cybersecurity framework, which consists of a set of activities managed by the framework that could be integrated within the automotive business process. The business process aims to present a particular set of tasks and activities over a particular time to reach a

business goal [JKM+17]. Therefore, the proposed framework aims to ensure the correctness of the applied security requirements and address the existing security issues as presented in this chapter context.

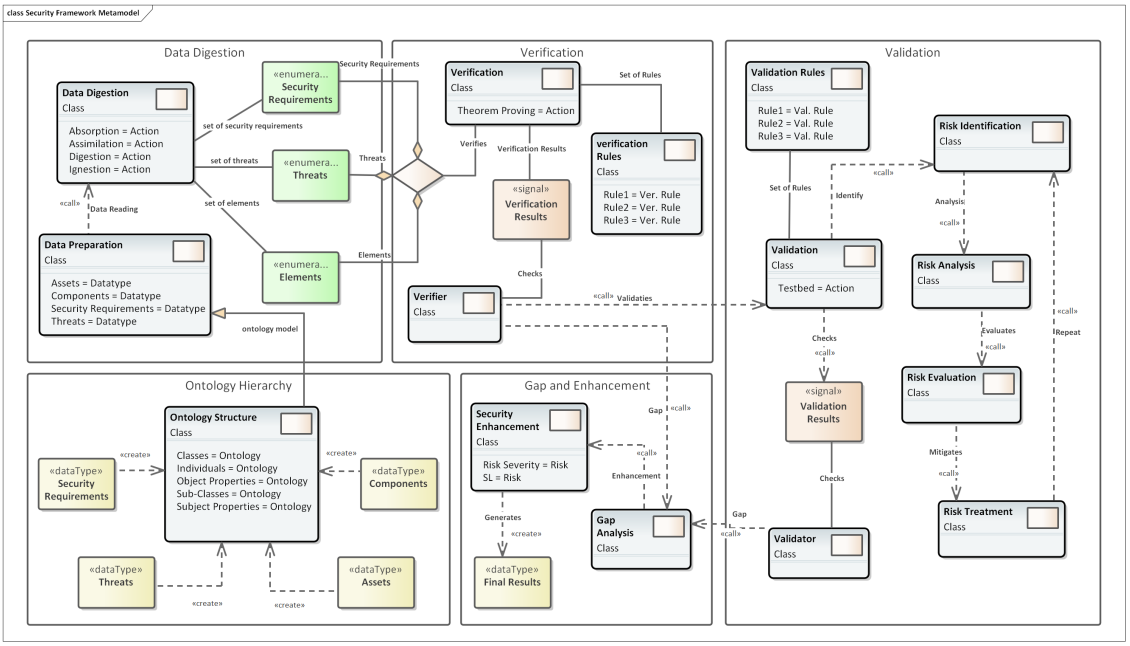


Figure 6.1: The metamodel of the ontology-based cybersecurity framework

As depicted in Figure 6.1, the first phase represents the data preparation that discussed in Chapter 4 and Chapter 5. The ontology structure shall be developed and appropriately defined to avoid any mistakes on the results regarding the poor structure. Then the framework accepts the ontology design of the vehicular design. The data designation phase aims to read and understand the contents of the ontology model to define exiting potential threats, security requirements, and vehicle elements. The verification phase collects all data and applies a set of rules to verify the logical correctness of the applied security requirements against exiting security threats. The verified data is delivered to the validation phase to test the correctness of the applied security requirements and check their effectiveness against existing potential threats. The identified security gaps are collected by the gap and enhancement phase to determine these gaps and suggest a new set of security requirements that can fill these gaps out. The following sections discuss more details of the activities and roles for the multiple phases in this proposed cybersecurity framework.

6.2 Data Digestion

The digestion phase reads and scans all input data in ontology form, representing all vehicle entities. As discussed in Chapter 4 and Chapter 5, all data is represented in an ontology format to provide a simplified data representation approach and provide a more efficient methodology for data analytics and processing. These data contain various classes, and sub-classes reflect the vehicle's main data categories. The BigML [1] discusses several stages (i.e., Ingestion, Digestion, Absorption, and Assimilation) of the big data to be pre-processed before manipulated by a system [Jos12].

[1] https://bigml.com

These stages are integrated into this work to manipulate the input data that will be processed by the other phases of the proposed framework. The pre-processing data phases are proposed to be combined with this work to obtain the necessary knowledge needed in the verification and validation process to validate vehicle security criteria against specific security vulnerabilities. These phases are described as follows:

- **Ingestion:** data collection as follows:
  - a list of all relevant details (i.e., name, type, security mitigation, etc.) on the components and assets of a vehicle,
  - all threats listed with all related information (i.e., name, Id, type, classification (category) and level of risk),
  - all related security requirements with the identified security weaknesses in the system model.
- **Digestion:** process original data in a formal language, which allows multiple values to be derived from the raw format.
- **Absorption:** extracts from the input all the data values needed to create an ontological representation.
- **Assimilation:** acts as a buffer to eliminate any unnecessary information. For example, threats with a low level of risk do not seem to be major security risks to a vehicle.

The input data could be classified into three main categories (i.e., components/assets, potential threats, and security requirements). The other data categories, such as security measures, security level, and risk severity, could be described as a part of these three main categories.

Components Identification: the components are selected based on the most common threats list, which is defined by the UNECE CS/OTA ad hoc "Threats 2" in [UNE17b]. This document is used in this study to identify potential vehicle threats and describe the basic vehicle components used in vehicle design. In addition, the assets are selected according to the V2X HSM (Vehicle-to-anything Hardware Security Module) component as discussed in [Car19] by "Car 2 Car Communication Consortium" [2]. There are multiple classes and subclasses are created to define the classification of vehicular components/assets in this work, as shown in Figure 6.2.

Algorithm 1 describes the outline procedures for reading and scanning all components and assets with all relevant information from the ontology hierarchy.

Threats Identification: identify the potential threats is the second step in the digestion phase process, which aims to define a list of all previously detected by any threat analysis methodologies (e.g., ThreatGet [3]). This phase scans all hierarchical ontology representation and understands all details relevant to threats entities. Figure 6.3 shows a set of potential threats that impact the "ECU" unit.

Algorithm 2 described the main outlines for reading and scanning all entities in the threats hierarchy to check the selected threats as negative consequences against components and assets.

[2] https://www.car-2-car.org
[3] https://www.threatget.com/

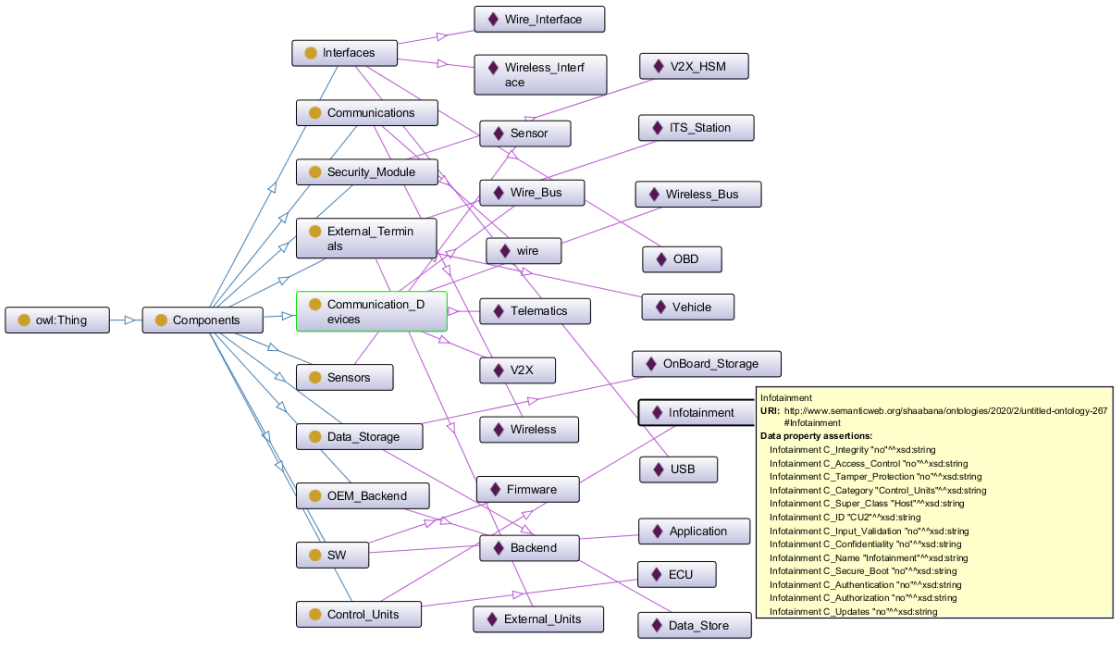


Figure 6.2: The classes and sub-classes of vehicular units (i.e., individuals)

Data: Reading all components/assets
Result: list of components/assets with related information
initialization;
Read the components/Assets ontology hierarchy;
while !isEmpty(Components_Assets) **do**
    Read ID;
    Read Name;
    Read Category;
    Read Division;
    if hasThreats **then**
        go to threats hierarchy and get all related threats to components/assets, as described in Algorithm 2;
    end
    while !isEmpty(Security_Measures) **do**
        Get all related security measures to a particular component or asset;
    end
    while !isEmpty(Security_Requirements) **do**
        Get all already selected security requirements for a particular component or asset;
    end
end

Algorithm 1: Scans all components and assets with retrieving all related data

Security Requirements Identification: it aims to read all the specified security requirements in the ontology entities, which are already specified to address the existing cyber risks regarding the potential threats identified. Figure 6.4 illustrates the scenario a set of potential threats (left)

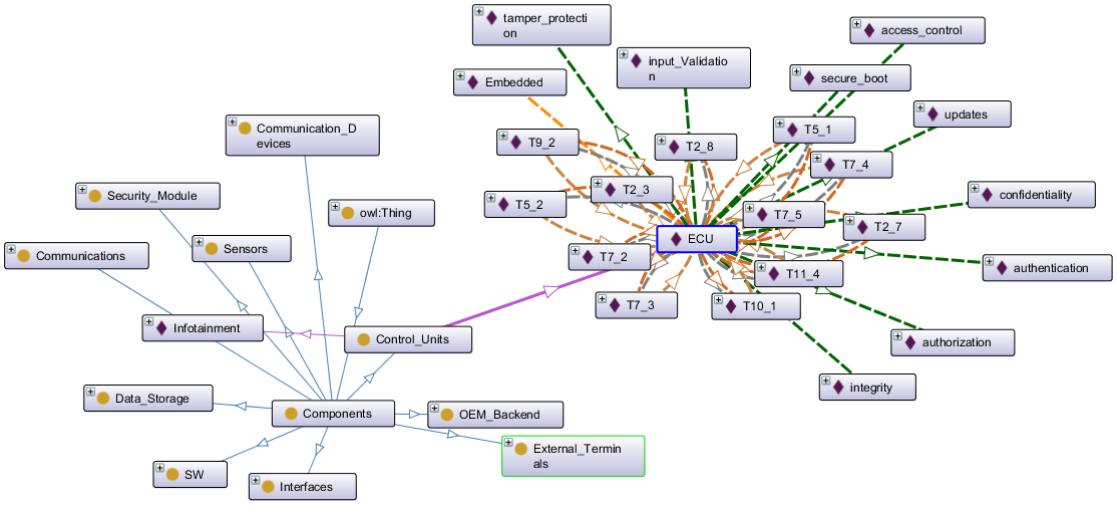


Figure 6.3: A set of potential threats that impact the ECU unit

Data: Reading all potential threats
Result: list of threats with related information
initialization;
Read the threats ontology hierarchy;
while !isEmpty(Threats) **do**
    Read ID;
    Read Name;
    Read Category;
    Read Likelihood value;
    Read Impact value;
    Read Severity estimated value;
    if security requirements **then**
        go to security requirements hierarchy and get all related data according to the selected security requirements used to address a particular threat, as described in Algorithm 3;
    end
    while !isEmpty(affected components/assets) **do**
        Get all components or assets, that this threat has a direct or indirect impact;
    end
    while !isEmpty(exploits) **do**
        Get all related security measures that a threat can exploit to pave the way to the attacker to attack the vehicle;
    end
end

Algorithm 2: Scans all threats with retrieving all related data

and a group of security requirements (right) are selected to address existing security issues and protect vehicular elements.

The figure depicts three main parts, these parts represent the problems (potential threats),

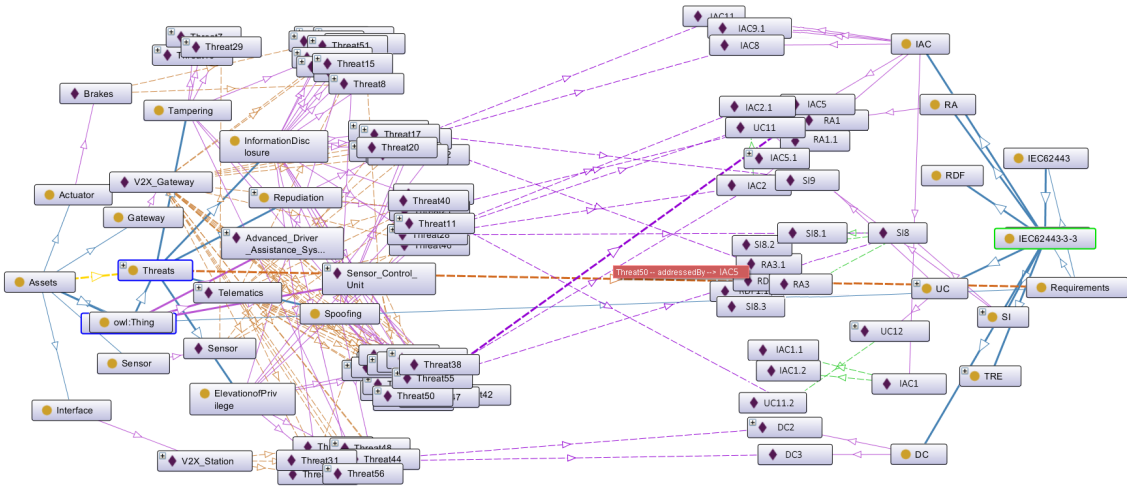


Figure 6.4: A set of potential threats against a set of security requirements are selected to fill security weaknesses

solutions (security requirements), and the matching between them:

- **Potential Threats (left-side):** hierarchy of all the vehicular components that are used in this case study. Besides, the identified potential threats that are detected by the ThreatGet tool.
- **The Security-Requirements(right-side):** represents a hierarchy of all security requirements that are selected to handle the detected potential threats.
- **The Matching between the Two Ontologies (middle):** these links represent specific security requirements are address one or more potential threat(s). The ontology on the left-side (threats) can be linked to a subtree on the right ontology (security requirements) that indicates a set of similar security requirements that can address a potential threat for handling related security issue [SMH+17], [SMM16].

Algorithm 3 outlines the main steps for reading and scanning all entities from the ontology hierarchy in order to understand the relationships between the selected security requirements and the identified potential threats.

6.3 Logical Verification: Theorem Proving

The verification phase behaves as a peer assessment to check almost every node in the ontology design to verify the logical correctness of all vehicle entities in the ontological form. The framework checks the consistency of the ontology design before performing the validation process. All entities in the ontology model are connected through determined relationships and are defined as an object property. For example, each component has a set of security measures used to mitigate the risk from the components with the existence of the potential threats. Logical verification based on the theorem proving is used in this work to logically check the consistency of the model design and detect the existing gaps. The logical verification's primary goal is to ensure the logical correctness of the ontology entities within the ontology structure. The verification is a step before performing the validation process to ensure the correctness of the security requirements against

```
Data: Reading all security requirements
Result: list of threats with related information
initialization;
Read the threats ontology hierarchy;
while !isEmpty(Threats) do
    Read ID;
    Read Name;
    Read Division;
    Read Levels;
    Read Properties;
    while !isEmpty(Properties) do
        Get all security properties are defined as security measures to be used to protect
          components/assets and address particular threats;
    end
end
```

Algorithm 3: Scans all security requirements with retrieving all related data

the previously identified threats. The theorem proving is one of the commonly used verification approach in the Cyber-Physical Systems - CPS based on mathematical reasoning as discussed in [BBG17]. These reasoning formulas can be defined in the SQWRL language and applied to the ontology model efficiently. Therefore, the theorem proving is integrated within this research context. In this study, the triples are specified to explain the relations between different entities in this proposed cybersecurity framework. Multiple object properties are defined to explain the relationship between entities in the ontology. The following is a sample of established properties, as shown below:

1. $T \overset{\text{addressedBy}}{\longrightarrow}$ SR; "T is the set of threats, and SR is the set of security requirements selected to address these security issues."

2. $C \overset{\text{protectedBy}}{\longrightarrow}$ SR; "C is the set of vehicular components, and SR is the set of security requirements selected to protects these elements from multiple potential threats."

3. $T \overset{\text{hasRisk}}{\longrightarrow}$ R; "T is the set of potential threats, and R is the severity risk level for each identified threat."

Throughout the ontology architecture, these statements are used to describe relationships between domain and range entities. For example, the following formulas explain the formal description of these statements:

- $T \overset{\text{addressedBy}}{\longrightarrow}$ SR: "Any threats need to be addressed by security requirement(s)"
  - $\forall_t \in D_{\text{Threat}}$,Threat (t) : t belongs to the domain "Threat".
  - $\forall_{sr} \in D_{\text{SecurityRequirement}}$,SecurityReq$(sr)$: sr belongs to the domain "Security_Requirement"
  - Threat(t): "t is threat"
  - Security_Requirement(sr): "sr is security requirement"
  - addressedBy(t, sr): "t needs to be addressed by a sr"

  - $\forall_t, \exists_{\geq sr}$ addressedBy (t, sr) "All threats must be addressed by one or more security requirements"

- $SR \xrightarrow{\text{protects}}$ C: Security requirement(s) protects vehicular components"
  - $\forall_{sr} \in D_{\text{SecurityRequirement}}$,SecurityReq$(sr)$: sr belongs to the domain "Security_Requirement"
  - $\forall_c \in D_{\text{Component}}$,Component (c) : c belongs to the domain "Component".
  - Component(c): "c is component"
  - Security_Requirement(sr): "sr is security requirement"
  - protects(sr, c): "sr must protect a c"
  - $\forall_c, \exists_{sr}$ protects (sr, c) "Security requirements are used to protect components"

- $T \xrightarrow{\text{hasRisk}}$ R: "Any threats have at least one risk level"
  - $\forall_T \in D_{\text{Threat}}$,Threat (T) : t belongs to the domain "Threat".
  - $\forall_r \in D_{\text{Risk}}$,Risk$(sr)$: r belongs to the domain "Risk"
  - Threat(t): "t is threat"
  - Risk: "r is a risk"
  - hasRisk(t, r): "t has r"
  - $\forall_t, \exists_{\geq r}$ hasRisk (t, r) "any identified threats must has at least one risk level."

Each potential threat shall have a relationship with at least one component with relevant security properties that have been exploited by that threat. This phase uses SQWRL language for performing the verification process and for checking the relation among different entities in the ontology. For instance, **"Any component affected by a threat needs to be protected by a security requirement; these requirements address exiting threats"**, this statement is translated into a series of mathematical operations performed on ontology to test the coherence (true or false) of entities and relationships:

- $\forall_T \in D_{\text{Threat}}$,Threat (T)
- $\forall_c \in D_{\text{Component}}$, Component (c)
- $\forall_{sr} \in D_{\text{SecurityReq}}$,SecurityReq$(sr)$
- Component(c): c is component
- Threat(t): t is threat
- protects(sr, c): "c needs to be protected by a sr"
- affectedBy(c, t): "c affected by threats"
- $\forall_c \{\exists_t [$ affectedBy $(c, t)] \rightarrow [\exists_{sr}$ protectedBy(c, sr)] $\}$
- $\forall_t \{\exists_{sr} [$ addressedBy $(t, sr)] \rightarrow [\exists_{sr}$ protectedBy(c, sr)] $\}$

The verification process starts with checking the consistency of relationships between the ontology entities to define the exact gap in the ontology structure. For example, each component has a set of security measures, that are used to mitigate the risk from the components against the existence of the potential threats, this scenario can be defined as **"ECU is a component and authorization is a security measure, and the ECU has authorization as a mitigation"**. In order to check the correctness of the relationship between components and related security measures, a propositional logic is used to describe it and estimate if this statement is true or false according to the logical consistency between ontology entities. The above statement can be described as follows:

- $\forall_T \in D_{\text{Threat}}$,Threat (T)
- C: "ECU is a component"
- SM: "authorization is a security measure"
- C $\rightarrow$ SM: "the ECU has authorization as a mitigation"
- C $\wedge$ SM $\wedge$ (C $\rightarrow$ SM): "ECU is a component and authorization is a security measure, and the ECU has authorization as a mitigation".

The following Table 6.1 demonstrates the logical verification of the above statement to check the consistency of the model design logically and defines the model design gaps in this work.

Table 6.1: The truth table of the hasMitigation relationship between components and security measures

C	**SM**	C $\wedge$ SM $\wedge$ (C $\rightarrow$ SM)
F	F	F
F	T	F
T	F	F
T	T	T

The truth table shows the logical steps of the verification process for checking the relationship "hasMitigation" between the component (e.g., ECU) and security measure (e.g., authorization). The result is true when the "hasMitigation" relationship between p (i.e., component) and q (i.e., security measure) is satisfied; where other results are false when any of p or q is false. Figure 6.5 illustrated a simple representation of the "hasMitigation" relationship between ECU as a component and the authorization security measure.

The SQWRL language is applied to create logical rules to check the logical consistency between the ontology entities. Algorithm 4 shows how the logical language is applied to the ontology to evaluate the correctness of relationships among entities to use these results later within this work in order to validate security measures in the vehicular design.

As shown in Figure 6.5, the ECU is an instance in the "Control Unit" subclass, and the authorization is an instance in the "Security_Measure" class. The SQWRL language can directly reach all instances of a class and all relevant subclasses, similar as in "Components(OnSecta1:ECU)" clause. The language gives the flexibility to navigate deeply into all entities within the superclass (Components) and check all other subclasses to check the SQWRL statement's satisfaction. Also, the same with the authorization within the "Security_Measure" class, as described in the SQWRL

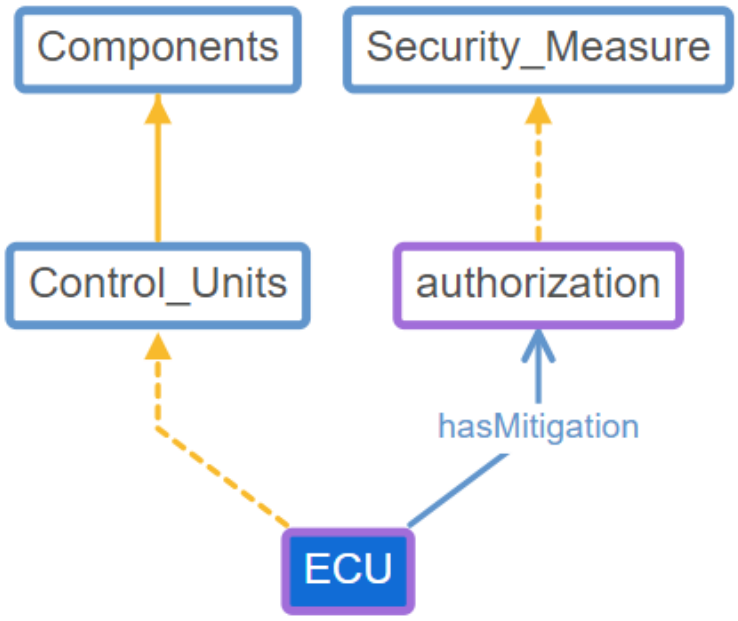


Figure 6.5: Relationship between components class (e.g., ECU) and the security measure (e.g., authorization)

Data: set of ontology entities
Result: A true value, which represents the correctness of relationships, or empty, which represents unsatisfied entities or relationships
initialization;
predicate := define the required predicate between the subject and object entities in the ontology hierarchy;
prefix := define the main ontology name-space as an ontology prefix of this ontology hierarchy;
while !isFound(continue) **do**
    subject:= define the required subject entity in the ontology hierarchy;
    subject_category := define the category type of the subject entity;
    object:= define the required object entity in the ontology hierarchy;
    object_category := define the category type of the object entity;
end
$SQWRL\_Rule := prefix : subject\_category(prefix : subject) \wedge prefix : object\_category(prefix : object) \wedge prefix : predicate(prefix : subject, prefix : object) \rightarrow sqwrl : select(true)$
Check_results := Apply the sqwrl statement into the reasoner engine (SQWRL_Rule);
return (Check_results);

Algorithm 4: The general structure of the SQWRL statement for checking the consistency between entities of a particular relationship

snippet. Afterwards, the relationship "hasMitigation" between the ECU and the authorization is checked to ensure the correctness of this relationship. The statement will return "true" in case the

statement clauses are correct; otherwise, the statement will return "nothing," which represents the result is "false" to indicate the incorrectness of this statement. In order to simulate this process, the UPPAAL check tool is used to demonstrate the logical verification process in this work; this tool is used in real-time systems for modeling, verification, and validation. It supports systems that can be modeled as a finite state machine with real-valued clocks. The graphical interface of the tool supports modeling, simulation, and verification. Also, the tool supports a stand-alone verifier for different applications, such as batch verification [UPP19]. Figure 6.6 illustrates the main structure of the UPPAAL verifier model for verifying the consistency between components and security measures.

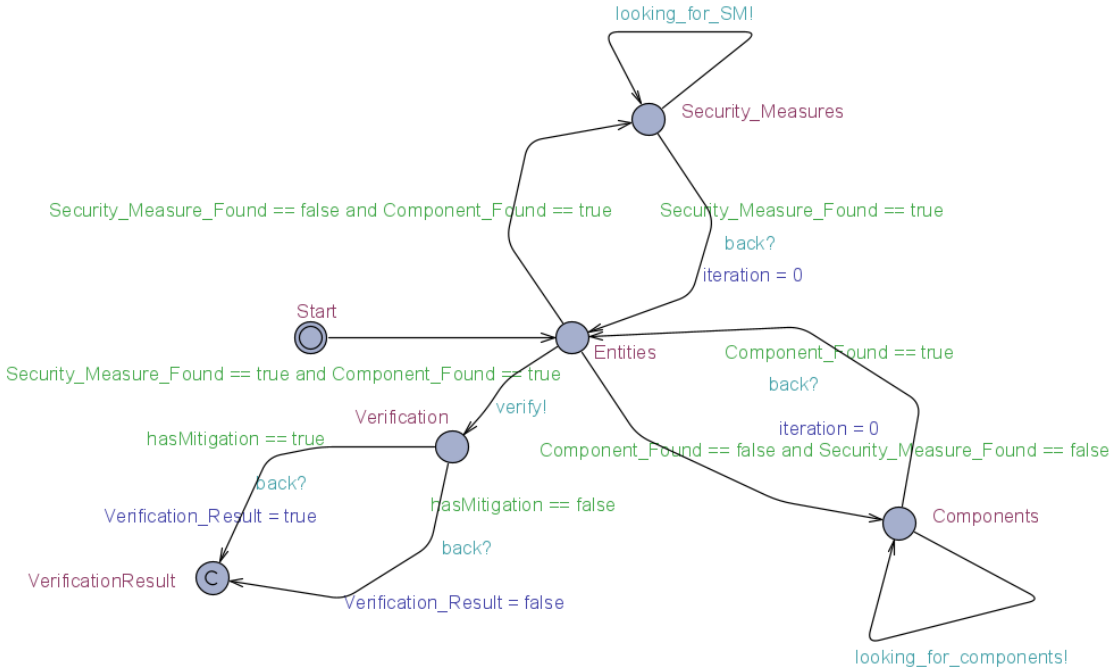


Figure 6.6: The main structure of the verification model for verifying the relationship between component and security measure

The model starts by calling the "componentChecker" sub-model checker (i.e., see Figure 6.7) to find and verify the target component among all entities in the ontology model. Once the component is found, the "Component_Found" flag will be true. Then the verifier model will call another sub-model checker called "SM_Checker" to find and verify the target security measure, as shown in Figure 6.7. The "SM_Checker" (i.e., see Figure 6.8) sets the Security_Measure_Found flag to true in case the target security measure is detected.

Afterwards, the main model calls C_SM_ModelVerification to verify the relationship between the identified component and related security measures. This verifier checks the updated flags (i.e., "Security_Measure_Found" and "Component_Found"); if both are true, then will check the "hasMitigation" relationship between them to verify the correctness consistency between the component and relevant security measure, as illustrated in Figure 6.9.

Finally, the checker sets the "Verification_Result" flag to true, when consistency between the entities is verified. Otherwise, the flag will be set to false, which represents the unsatisfied result. The framework will handle the unverified findings to fill out the exiting gap in the vehicle ontology design. Meanwhile, the hasMitigation relationship is verified; the verification process continues

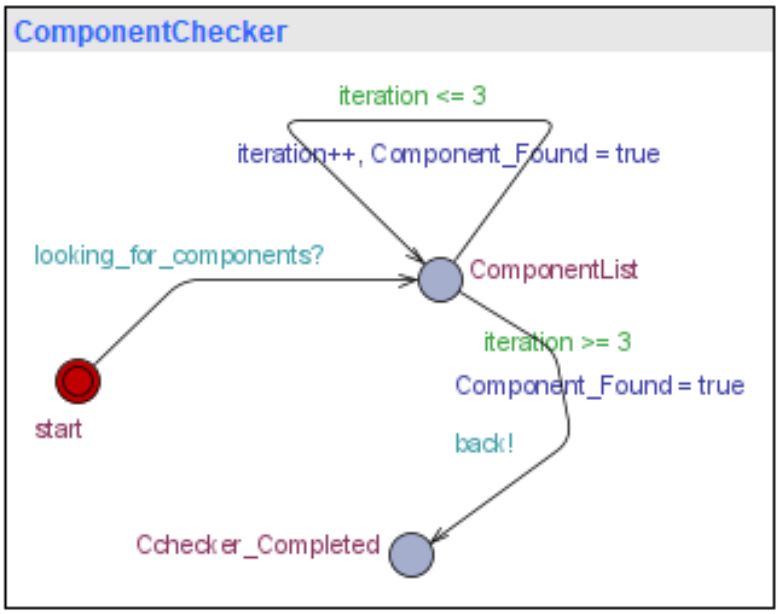


Figure 6.7: Component model checker

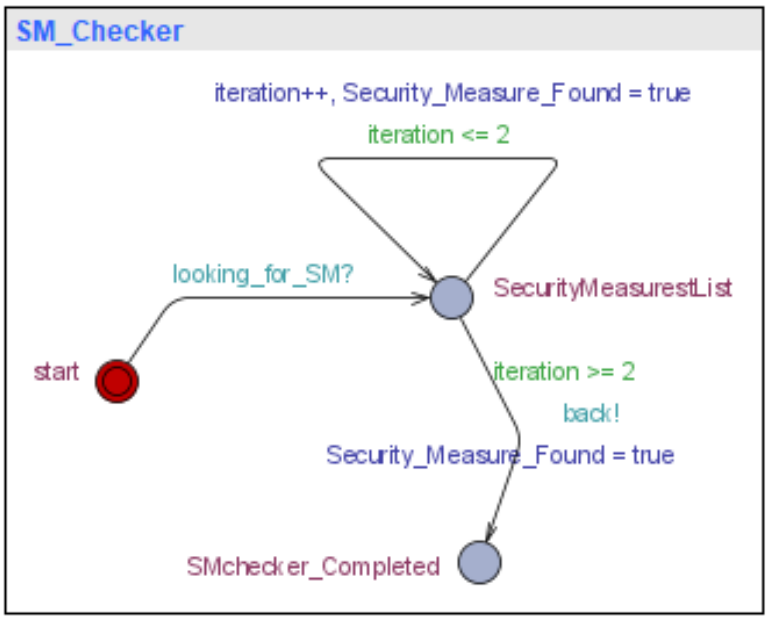


Figure 6.8: Security measure model checker

to verify the other relationships among all entities in the ontology model. The specified security measure, which has a relationship with a component, is considered the main next step that the verifier follows to check the correctness between this security measure and other entities. Any malicious activity seeks to damage, manipulate, compromise, or abuse of rights is called a threat. The threat has a potential impact on a particular target, such as components or valuable assets such as sensitive data. Also, it aims to exploit existing security weaknesses in the system design, to

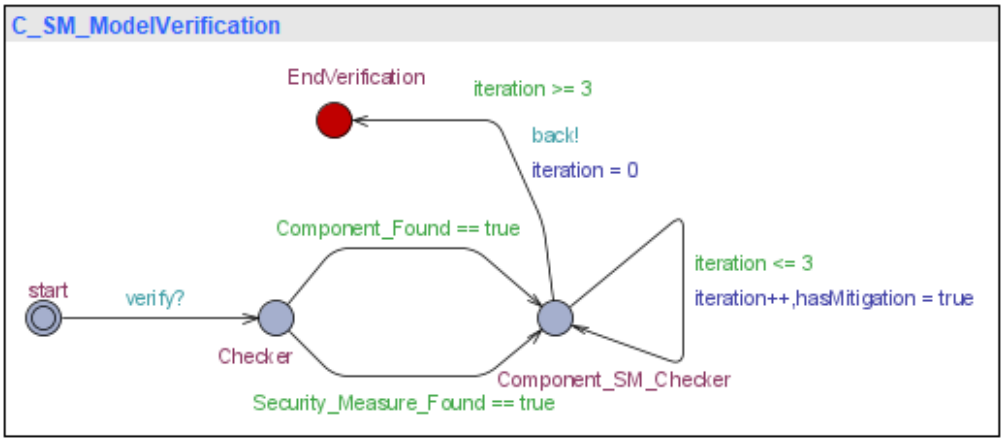


Figure 6.9: The hasMitigation relationship between the component and relevant security measure

pave the way into the attacker to start performing alternative ways of malicious actions. Therefore, in this work, we use the *exploits* relationship to define the relation between a threat and security measure because once a threat succeeds in exploiting a particular security measure, in this case, the component contains a security flaw that needs to be addressed by more security concerns. Then this threat, which exploits a particular security measure, can have a negative impact on a specific component. Therefore, an *impacts* relationship is defined to represent the relationship between a particular threat and relevant affected component. Also, the same security measure is a part of a set of security conditions that need to be achieved to reach a particular security protection level. These conditions are called security requirements; therefore, a particular security measure has a relation(s) with one or multiple security requirements. Furthermore, *hasProperty* relationship is defined to represent the connection between the security measure and relevant security requirements. Figure 6.10 illustrates the relationships between component, threat, security measure, and security requirements, which need to be checked by the model verifier to ensure the correctness of the consistency of these relationships between the proper entities.

The figure shows that a threat "T7_4" exploits the authorization security mechanism that is selected in order to be the security mitigation of the component "ECU"; therefore, the threat T7_4 "impacts" the component "ECU". The authorization security measure is "partOf" a security requirement "EDR_3_11_1" and another relationship "hasProperty" explains that the security requirement has an authorization as a security property. The following statement represents this scenario in propositional clauses:

- Each component has a security measure as a security mitigation: $\exists_{C,SM}[\,\text{hasMitigation}\,(\text{C},\text{SM})]$
  - $\forall_C \in D_{\text{Components}}$,Components (C)
  - C: "ECU is a component"
  - $\forall_{SM} \in D_{\text{Security\_Measures}}$,Security_Measure (SM)
  - SM: "Authorization is a security measure"
  - hasMitigation(C, SM): "C (i.e., ECU) has a security mitigation SM (Authorization)"
  - *hasMitigation*: $(\text{C} \rightarrow \text{SM})$ "ECU has authorization as a mitigation"
  - a truth table is generated to check the consistency correctness of this relationship, as is described as follows (Table 6.2):

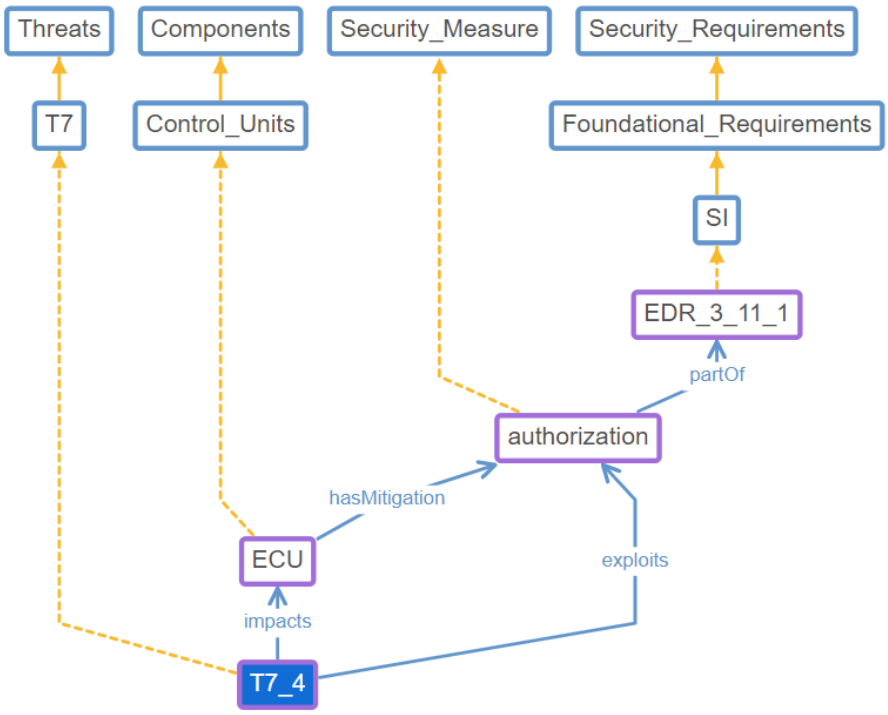


Figure 6.10: Relationship between a component, a relevant security measure, a potential threat, and a security requirement

Table 6.2: hasMitigation relationship truth table

C	SM	$(C \wedge ((C \rightarrow SM) \wedge SM))$
F	F	F
F	T	F
T	F	F
T	T	T

– An SQWRL language code is applied to test the logical consistency between the various ontology entities in this "hasMitigation" relationship, as described in Listing 6.1. The result would be either "true" when the this relationship is satisfied, or "false" in case of this relationship or the relevant entities are not satisfied. An SQWRL snippet is described as follows:

```
1 OnSecta1:Components(OnSecta1:ECU) ^ OnSecta1:Security_Measure(OnSecta1:
    authorization) ^ OnSecta1:hasMitigation(OnSecta1:ECU, OnSecta1:
    authorization) -> sqwrl:select(true)
```

Listing 6.1: The "hasMitigation" relationship between the ECU (component) and the authorization (security measure)

- Each threat could exploit a security measures such as authorization, which is considered as security vulnerability: $\exists_{T,SM}[\text{ exploits } (T, SM)]$

- $\forall_T \in D_{\text{Threat}}$,Threat (T)
- T: "T7_4 is a threat"
- exploits(T, SM): "Threat (T7_4) exploits security vulnerability (Authorization)" (e.g., the exploited security measure is considered as a security vulnerability in the system model)
- *exploits*: $(\text{T} \rightarrow \text{SM})$ "T7_4 exploits the authorization security measure"
- a truth table is generated to check the true or false of the consistency of this relationship, as is described as follows (Table 6.3):

Table 6.3: exploits relationship truth table

T	**SM**	$(\text{T} \wedge ((T \rightarrow SM) \wedge SM))$
F	F	F
F	T	F
T	F	F
T	T	T

- The SQWRL language code is applied to test the logical consistency between the various ontology entities in this "exploits" relationship, as described in Listing 6.2. An SQWRL snippet is described as follows:

```
1  OnSecta1:Threats(OnSecta1:T7_4) ^ OnSecta1:Security_Measure(OnSecta1:
     authorization) ^ OnSecta1:exploits(OnSecta1:T7_4, OnSecta1:
     authorization) -> sqwrl:select(true)
```

Listing 6.2: The "exploits" relationship between the T7_4 (threat) and the authorization (security measure)

- Each security measures is considered as a part of a security requirement: $\exists_{SM,SR}[\text{ partOf } (\text{SM}, \text{SR})]$
  - a truth table is generated to check the consistency correctness of this relationship, as is described as follows:
  - $\forall_{SR} \in D_{\text{Security\_Requirements}}$,Security_Requirements (SR)
  - SR: "EDR_3_11_1 is a security requirement"
  - partOf(SM, SR): "SM (Authorization) is a part of a security requirement (EDR_3_11_1)"
  - *partOf*: $(\text{SM} \rightarrow \text{SR})$ "a security measure is part of security requirement(s)"
  - a truth table is generated to check the true or false of the consistency of this relationship, as is described as follows (Table 6.4):
  - The SQWRL language code is used to test the logical consistency between the various ontology entities in this "partOf" relationship, as described in Listing 6.3. A SQWRL snippet is described as follows:

Table 6.4: partOf relationship truth table

SM	SR	$(SM \wedge ((SM \rightarrow SR) \wedge SR))$
F	F	F
F	T	F
T	F	F
T	T	T

```
OnSecta1:Security_Measure(OnSecta1:authorization) ^ OnSecta1:
    Security_Requirements(OnSecta1:EDR_3_11_1) ^ OnSecta1:partOf(OnSecta1:
    authorization, OnSecta1:EDR_3_11_1) -> sqwrl:select(true)
```

Listing 6.3: The "partOf" relationship between the authorization (security measure) and the EDR_3_11_1 (security requirement)

- Then, each identified threat can impact any component, which contains any security vulnerabilities (i.e., security measure): $\exists_{T,C}[\text{ impacts } (T,C)]$
  - impacts(T, C): "T (T7_4) impacts the ECU unit"
  - *impacts*: $(T \rightarrow C)$ "a threat (T7_4) impact a vehicualr component (ECU unit)"
  - a truth table is generated to check the true or false of the consistency of this relationship, as is described as follows (Table 6.5):

Table 6.5: impacts relationship truth table

T	C	$(T \wedge ((T \rightarrow C) \wedge C))$
F	F	F
F	T	F
T	F	F
T	T	T

  - The SQWRL language code is used to test the logical consistency between the various ontology entities in the "impacts" relationship, as described in Listing 6.4. An SQWRL snippet is described as follows:

```
<!-- http://www.semanticweb.org/shaabana/OnSecta.owl#
Security_Requirements -->
      <owl:Class rdf:about="http://www.semanticweb.org/shaabana/
OnSecta.owl">
     <owl:disjointWith rdf:resource="http://www.semanticweb.org/
shaabana/OnSecta.owl#Threats"/>
 </owl:Class>
```

Listing 6.4: The "impacts" relationship between the threat (T7_4) and the component ECU

- The overall relationship is checked to test the correctness of all previously discussed relationships between entities in the ontology model:
  - $\exists_{C,SM}[$ hasMitigation $(C, SM)] \rightarrow [\exists_{SM,SR}$ partOf(SM, SR)]
  - $\exists_{T,C}[$ impacts $(T, C)] \rightarrow [\exists_{T,SM}$ exploits(T, SM)]
  - a truth table is generated to check the true or false of the consistency of the overall discussed relationship, as is described as follows (Table 6.6):

Table 6.6: Truth table of all previously discussed relationship

C	T	SM	SR	$(C \wedge (T \wedge (SM \wedge (SR \wedge ((C \rightarrow SM) \wedge ((T \rightarrow SM) \wedge ((SM \rightarrow SR) \wedge (T \rightarrow C))))))))$
F	F	F	F	F
T	F	F	F	F
F	T	F	F	F
T	T	F	F	F
F	F	T	F	F
T	F	T	F	F
F	T	T	F	F
T	T	T	F	F
F	F	F	T	F
T	F	F	T	F
F	T	F	T	F
T	T	F	T	F
F	F	T	T	F
T	F	T	T	F
F	T	T	T	F
T	T	T	T	T

- The SQWRL language code is used to test the logical consistency between the various ontology entities in the all discussed relationships in the ontology design, as presented in Listing 6.5. An SQWRL snippet is described as follows:

```
1 OnSecta1:Components(OnSecta1:ECU) ^ OnSecta1:Security_Measure(OnSecta1:
    authorization) ^ OnSecta1:hasMitigation(OnSecta1:ECU, OnSecta1:
    authorization) ^ OnSecta1:Threats(OnSecta1:T7_4) ^ OnSecta1:exploits(
    OnSecta1:T7_4, OnSecta1:authorization) ^ OnSecta1:
    Security_Requirements(OnSecta1:EDR_3_11_1) ^ OnSecta1:partOf(OnSecta1:
    authorization, OnSecta1:EDR_3_11_1) ^ OnSecta1:impacts(OnSecta1:T7_4,
    OnSecta1:ECU) -> sqwrl:select(true)
```

Listing 6.5: Check the relationships among all previously discussed entities in the ontology model

According to this SQWRL snippet and the identified relationships in Figure 6.10, the SQWRL returns true, which represents the correctness of the selected entities with their relationships.

Figure 6.11 illustrates a true boolean result, which is returned by the semantic reasoner. The false results are handled by the gap analysis phase, which takes care of managing and filling out the existing security gap in the vehicular ontology design.

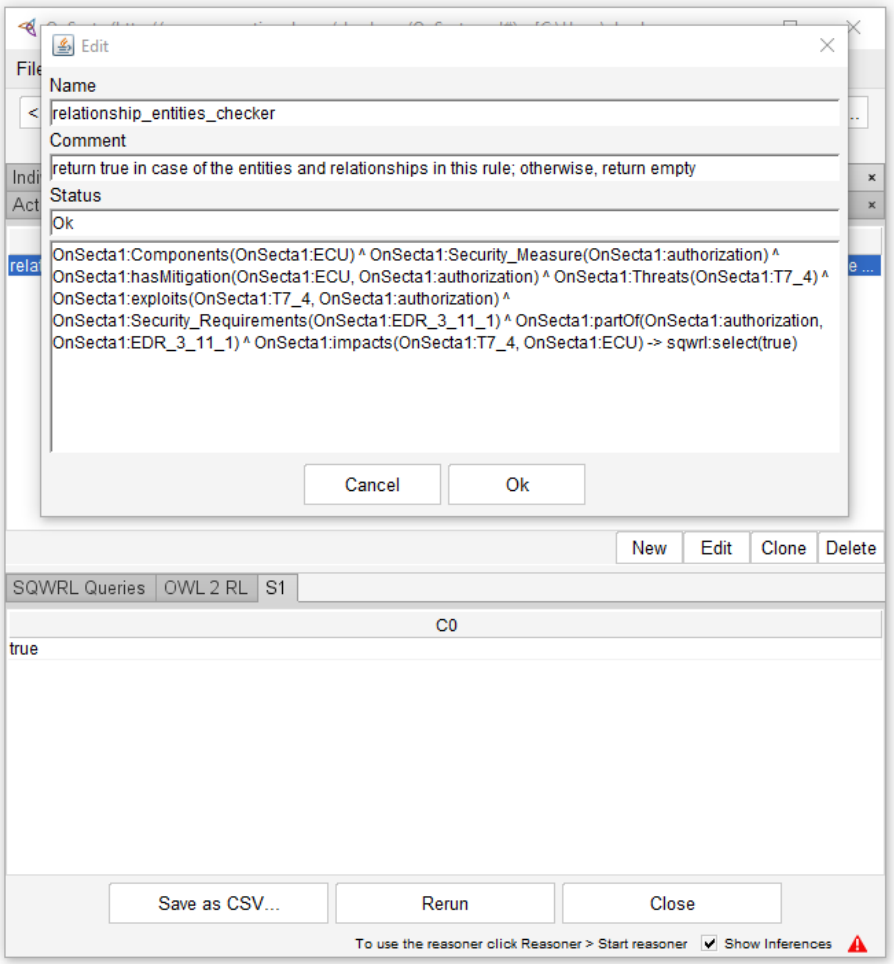


Figure 6.11: The returned boolean result of the semantic reasoner

6.4 Security Validation: Testbed Execution

The validation process aims to determine whether the identified security requirements are validated against existing security issues. This phase is based on the testbed execution to perform procedures for each vehicular element individually. This process aims to evaluate the correctness of the selected security requirements. The testbed approach consists of a set of processes inspired by the ISO risk management process, as discussed in detail in [iso11]. In order to build a secure system, it is essential to know potential threats that affect security protection mechanisms in the system model. It helps in selecting appropriate sets of security requirements that can address these security issues. In the validation process, the situation is similar; to check how secured a system model is, we need to test the applied security requirements' effectiveness against malicious activities. Therefore, we need to know potential threats in the system model to check applied security requirements against existing security issues.

In some cases, a vehicle ontology representation model could not contain threats, which is considered a challenging process, because no threats are defined to check the correctness of security requirements against them. ISO 27005 [iso11] represents different phases for managing risks, starting with risk identification, risk analysis, risk evaluation, and risk treatment. All of

these phases help to define risk and handle it by an appropriate set of security requirements. The same strategy needs to be followed by our proposed approach for identifying potential risks to test the effectiveness of the applied security requirements. Furthermore, the validation process is developed to follow the same strategy of the ISO risk management process. As presented in 7.3.1, the framework checks the correctness of the applied security requirements against potential threats compared to outcomes of the risk treatment phase. The framework is developed to be fully-adaptable for different ontology input forms. In some cases, potential threats are missing, and the framework needs to check the correctness of the applied security requirements. Therefore, the validation process tests the accuracy of security requirements against potential threats according to outcomes of the risk analysis phase, as presented in 7.3.3. In some other cases, there are no security requirements defined in the ontology input form. Therefore, the outcomes of the risk treatment phase in the validation process could be used to fulfil security gaps, as discussed in 7.3.2 and 7.3.4. In addition, the framework handles security gaps and improves the vehicular security level by integrating the outcomes risk treatment phase. Furthermore, using the validation process outcomes to handle security gaps helps saving time and avoids process repetitions. Figure 6.12 shows the main processes in the validation approach, which are applied to each vehicular component/asset to validate the applied security protection mechanisms. These security mechanisms aim to protect vehicular units from different cyber attacks.

Figure 6.12: The risk management phases in the validation process

These processes are adapted to fit in this work to create a new framework able to validate the selected security requirements against a wide range of potential threats, to protect particular component/asset. Figure 6.13 shows how these phases can be used to be integrated with the previously defined ontology structure as is depicted in Figure 3.3.

The figure describes the vehicle model's ontology levels with a collection of risk management activities, starting with risk identification and ending with risk treatment. These actions are described as a series of risk management actions in a perpendicular form with an ontology layer for illustrating the interaction between these layers and the proposed model. The proposed testbed execution validation technique is based on risk management actions to provide a holistic approach that can more efficiently identify, analyze, prioritize, and treat each vehicular component/asset individually. Figure 6.14 illustrates the main phases of the proposed security requirements validation process.

Security requirements address potential threats that are found to attack a component or asset in the vehicle. The validation process uses the security characteristics for each component and asset within the vehicular ontology model to test the matching accuracy between the identified threats and the selected security requirements for a particular vehicular unit. The rule engine plays a vital role in this process; it generates a set of rules for analysing the risks in the given ontology model and defines the exact security issue in the ontology design. According to the identified risks, the rule engine generates a set of logical rules based on a reasoning methodology to deduce the most applicable security requirements for a security issue. Then the framework automatically compares the obtained results (security requirements) with the existing ones, to check if the previously

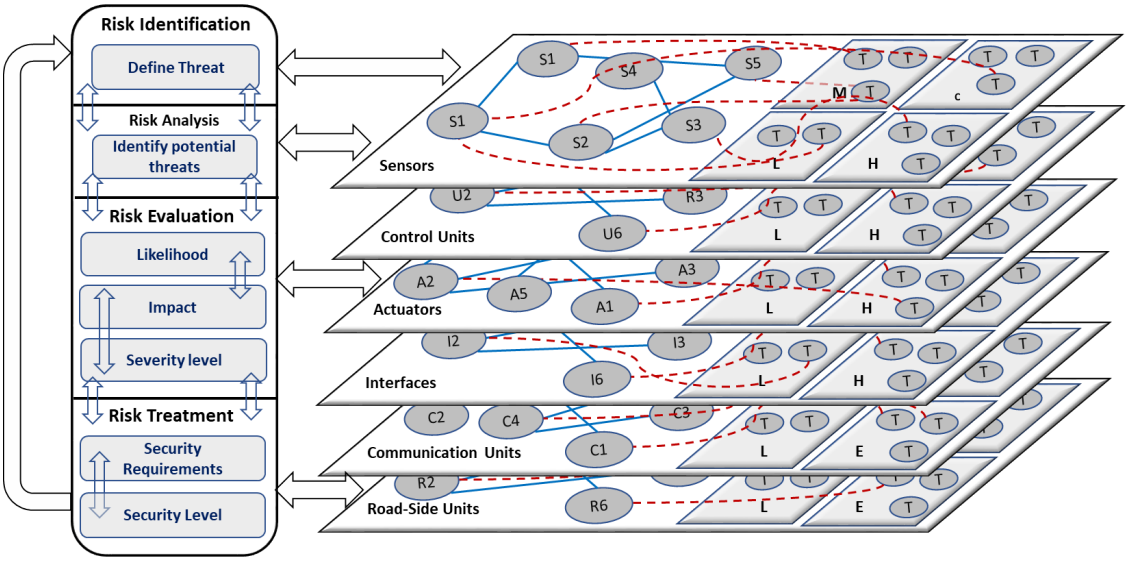


Figure 6.13: The risk management process with the vehicular ontology hierarchy

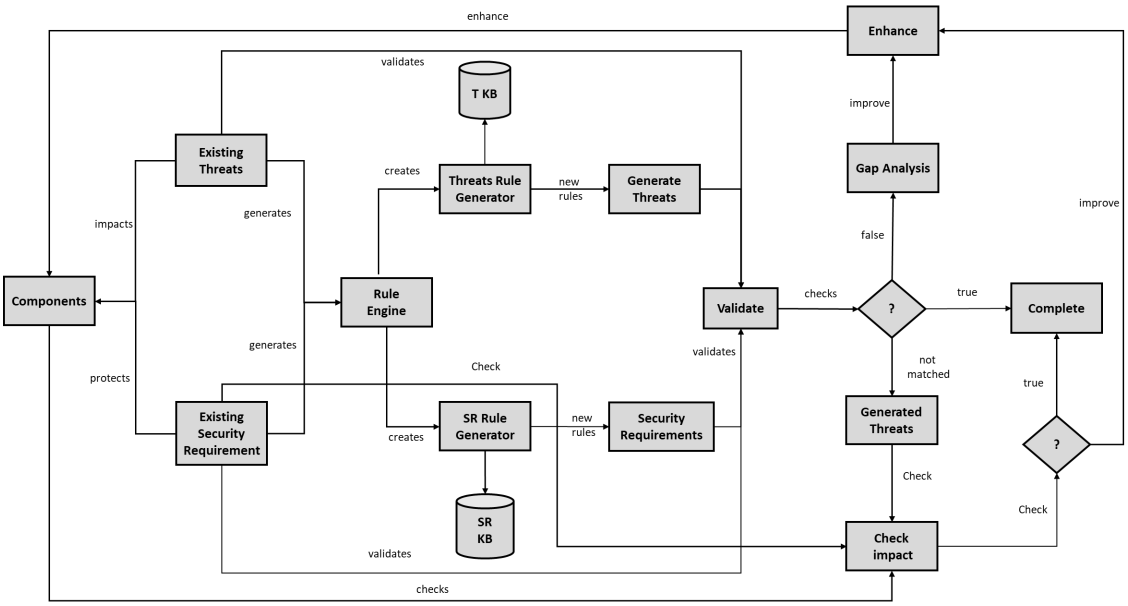


Figure 6.14: The main phases of the validation process

selected security requirements are applicable for addressing the existing security issues.

In some cases, the integrated security requirements within the ontology model could not completely match the inferred results. Therefore, the cybersecurity framework performs a reverse engineering action from the solution to the problem to deduce security issues from the assigned security requirements. Then it checks the impact of these security requirements on the inferred potential threats. The false results are collected to be handled by the gap analysis and security enhancement phases. The framework aims to define and highlight the existing security gaps in the vehicular design and fill out this gap by selecting the proper security requirements against the existing security issues. Besides, outcomes of the framework are generated as a suggested group

of applicable security requirements that matched the security characterises in the given ontology model, which assist the system designer in selecting and finding the most appropriate security requirements that help to fulfil the current security gaps in the vehicular design. The following sections discuss the main activities for each process in the validation approach.

6.4.1 Risk Identification

Risk identification is considered an essential step in the risk management process. This process has an active role in the whole risk management process because a correct risk identification process ensures an effective risk management process [Tch02]. The process of identifying threats is part of this research. The proposed framework provides a set of threats that could affect the vehicular system. The primary resources are used in this work to define a wide range of potential threats:

- UNECE Threat Intelligence for automotive Cybersecurity and Over-the-Air Software Update [UNE17b].
- Threats according to the V2X security module protection profile [Car19].
- CVE - Common Vulnerabilities and Exposures [Com20].

These resources are used to build a threat database, which contains a wide range of potential threats that can be used in the risk analysis process to identify a different range of threats that impact the vehicular system. The proposed work manages threats to perform a risk analysis to identify potential security vulnerabilities to be used in this work. This is called threat management, which aims to manage the threat database with all included threats. The main structure of the threat management is shown in Figure 6.15.

The "Rule Generator" performs a series of calculations to translate threat information into a form used in the analytical process. The rule engine receives threat information in a text form, as described in Chapter 5, and translates it into a version of SQWRL. Algorithm 5 outlines the main steps, followed by the rule generator for creating threat rules.

The threats identifier is responsible for extracting the threat rules from the knowledge-base (KB). These threats are collected to be used in the risk analysis process according to the vehicle input model. The threats KB is based on threat intelligence with all relevant details, such as category, definition, likelihood, impact, and the threat rule statement. Threat Intelligence or Cyber Threat Intelligence (CTI) analyzes all aspects of possible threats that affect an enterprise. The primary CTI purpose is to understand better the most common threats impacting an organization [SKI+19]. The threat knowledge-base is designed and developed to include the most common potential threats in the automotive domain. The primary source incorporated with this work is UNECE threats list as a threat intelligence for automotive cybersecurity and over-the-air software update [UNE17b]. In addition, several other threats are combined with this research from the V2X hardware security module protection profile [Car19]. Also, some selected threats are collected from the CVE [Com20] to identify some likely security vulnerabilities for the case study. More details about the case-study are discussed and explained in chapter 7. Threats used in this work are classified into 14 groups of threat categories. Figure 6.16 illustrates threats classification hierarchy, of threats that are integrated within the context of this research.

The following list defines threat categories that are created in this work:

- **T1**: Compromise of back-end server.

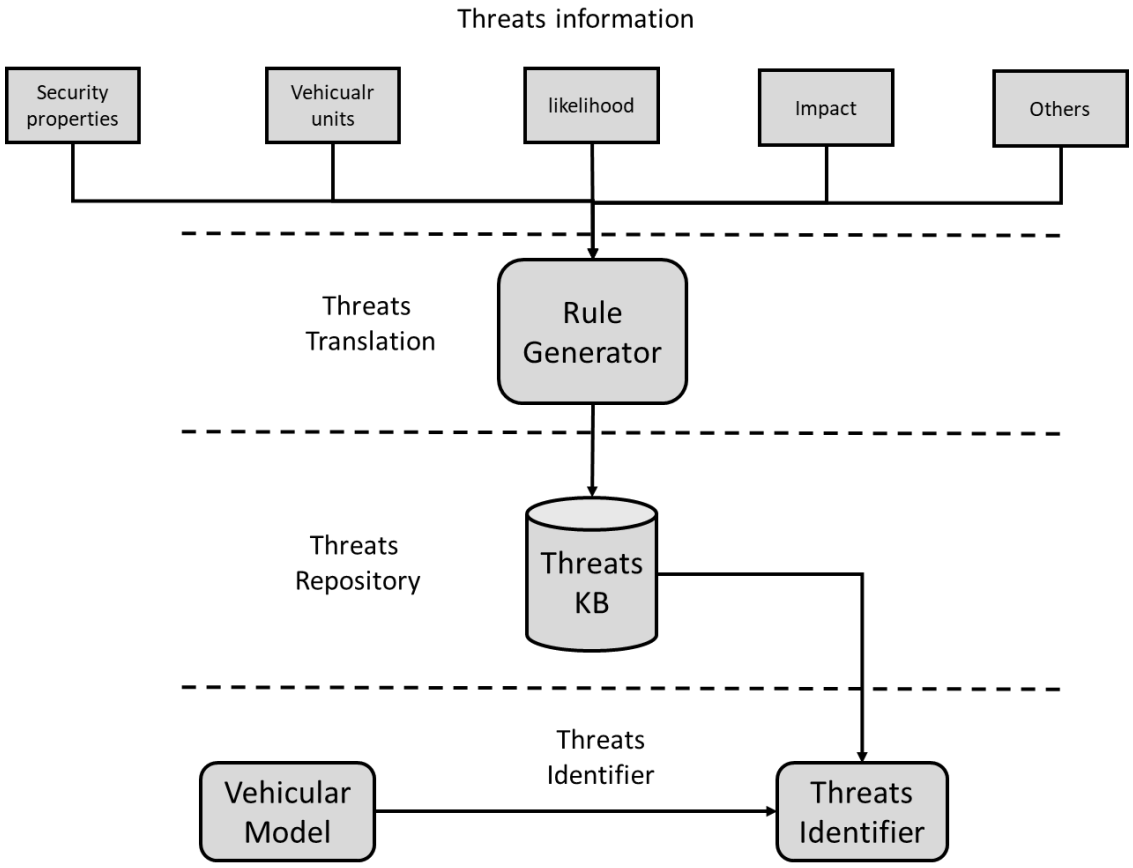


Figure 6.15: The structure of the threat management process

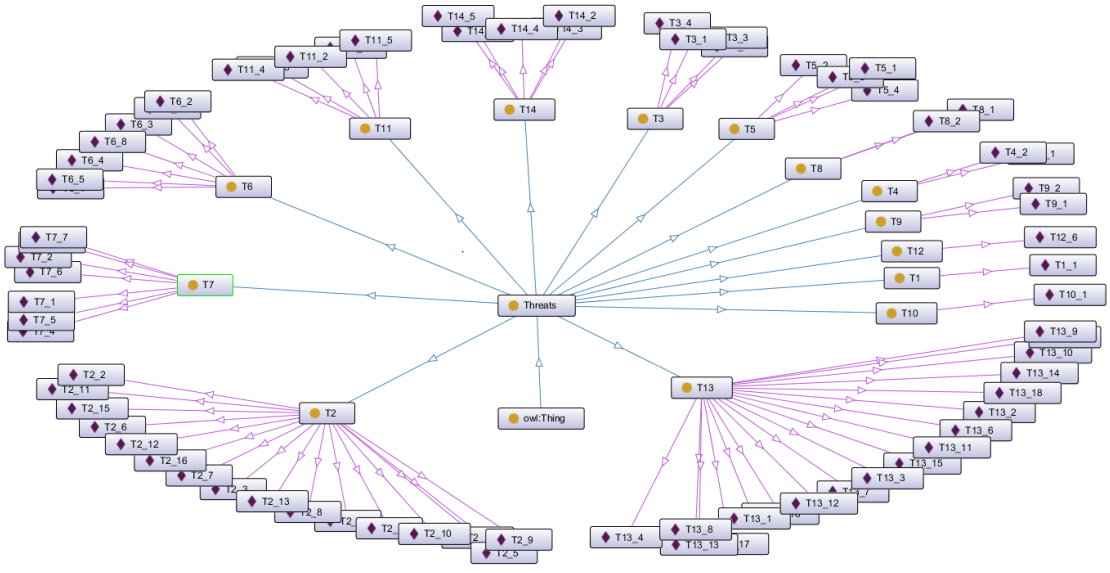


Figure 6.16: The potential threats class and sub-classes represent the threats categories

Data: Translating threats raw data into SQWRL form
Result: Set of SQWRL rules describe the threat structure
initialization;
Read the threats ontology hierarchy;
while !isEmpty(ThreatDataRecord) **do**
    Read ID;
    Read Title;
    Read Category;
    Read Risk_Level;
    Read Likelihood;
    Read Impact;
    Read Severity;
    Read exploits[];
    Read Impacts[];
    Read addressedBy[];
    ID, exploits[], Impacts[] = Filteration(); "select the threat information that are needed for creating SQWRL rules"
    while !isEmpty(AllThreats) **do**
        Set Threatindex;
        while !isEmpty(AllComponents) **do**
            Set Componentindex;
            Set Rule;
            $Rule = Threats(ID) \wedge Components(Impacts[Componentindex++])$;
            while !isEmpty(AllSecurityProperties) **do**
                Set Propertiesindex;
                $Rule = Rule + exploits(exploits[Propertiesindex++])$;
            end
            $Rule = Rule \rightarrow sqwrl : select(threat)$;
            $Store(Rule)$; "Store into the threat database";
        end
    end
end

Algorithm 5: Threats rule generator

- **T2**: Communication channels used to attack a vehicle.
- **T3**: Update process used to attack a vehicle.
- **T4**: Abuse of privilege.
- **T5**: Human factor and social engineering.
- **T6**: Compromise of external connectivity.
- **T7**: Target of an attack on a vehicle.
- **T8**: System design exploits (inadequate design and planning or lack of adaption).
- **T9**: Data loss/"data leakage" from vehicle.

- **T10**: Physical manipulation of systems to enable an attack.
- **T11**: Vehicle used as a means to propagate an attack Category agreed as out of scope due to vehicle needing to be compromised first.
- **T12**: Communication loss to/from vehicle (potentially out of scope as not cybersecurity).
- **T13**: V2X_HSM threats.
- **T14**: CVE threats.

The framework selects a set of threats that could affect a vehicular element, then apply previously threat rules to define exact potential threats that have an impact against a particular vehicular element. Applying these rules will be discussed in the following section.

6.4.2 Risk Analysis

One single vulnerable unit in a vehicle could impact all elements, or the whole vehicle is being exposed to a higher degree of cyberattacks. Furthermore, it is essential to understand the exact security weaknesses in a vehicle at the early stages of the security engineering process because once the vehicle is built, it becomes more difficult to add security. A risk analysis could be performed in varying degrees of detail such as assets, vulnerabilities, and incidents that have occurred [iso11]. The risk analysis is an activity that aims to define relationships between threats and vulnerabilities which threaten the vehicle. These relationships establish a set of classes and subclasses to decompose threat scenarios into possible attack paths [ISO20b]. This activity consists of two main stages as asset identification and threats analysis. The general risk framework was developed by the National Institute of Standards and Technology - NIST [4] workshop and proposed six concepts of the risk analysis:

1. **Assets:** are considered data, device, component, or either a physical or a logical object that has a value to an attacker. In this work, identifying assets plays an integral part in defining security weaknesses that an attacker could use to start performing malicious actions. It is essential to identify all components in a vehicle, to be used further in the other phases of the cybersecurity management process. This phase determines the security properties (security mitigations) of the defined components (such as secure boot, authentication, encryption, others). These properties are essential in the threat and vulnerability analysis processes and for selecting the most relevant security requirements to address the identified security weaknesses in a vehicle. Therefore, in this work, it is essential to know which threats could impact an asset. Furthermore, the asset identification process should be applied to collect more information about the vehicular asset, as discussed as follows [Ass13]:
    - create an asset record,
    - identify asset information,
    - define the topological structure of interconnected assets.
2. **Vulnerabilities:** no system can be built 100% of cyber secure, because the system users still need to get access into their information [Nor16]. An attacker exploits a vulnerable point

[4] https://www.nist.gov/

to get access into the system or its information. In the vehicular system the vulnerability analysis process is concerned with exploring, defining, identifying, and prioritizing vulnerabilities or security weaknesses in vehicles [SMS14]. Security mechanisms need to be used to avoid those threats that exploit existing vulnerabilities in system units. Security mechanisms are a set of different classes of defensive activities such as detective, preventive, corrective, etc. [MJGA15]. A summary of existing database's vulnerability is discussed in [SDK19]. Vulnerabilities can be found at hardware, software, or network level, an overview of a possible classification of vulnerabilities has been presented in [SGPS14].

3. **Threats:** are defined as a set of events that have the potential to cause a negative impact on asset, control system, or individual [IEC19]. In the vehicular domain, the threat analysis is an activity that identifies the potential negative actions that affect the security mechanism in vehicles. The threat analysis process can be divided into the following essential steps, as presented in [STH+19]:
   a) Model the vehicle with all security related assumptions and necessary information.
   b) Model potential adversaries with their tactics, techniques, and procedures.
   c) Apply the threat model to the system model to identify potential threats.
   d) Evaluate all identified threats and decide on the risk treatment.
   e) Update the system model with the security countermeasures.
   f) Repeat step "c" in order to identify missed or new threats.

4. **Impact:** defined as consequences or effects of risk. The impact assessment process is an activity that aims to evaluate risk when potential threats and security vulnerabilities are defined. In the automotive domain, it is important to ensure that different types of impacts do not damage the vehicle or cause other accident scenarios. It should be avoided that a cybersecurity attack [SLAMH18]:
   - causes immediate damage to the environment or human lives (**safety**),
   - causes loss of control over personal information (**privacy**),
   - causes financial damage (**finance**),
   - negatively impacts the operation and traffic flow (**operation**).

   Table 6.7 shows these four factors of impact (i.e., safety, financial, operational, and privacy) with their proposed numerical values. The safety category is based on ASIL levels. The Automotive Safety Integrity Level (ASIL) value represents the level of hazard. ASIL is designed to identify appropriate risk mitigation measures to avoid unacceptable risk. ASIL values are classified in four classes, including A, B, C, or D; where D is the critical level of safety, and A is the lowest one [Alh17]. We use 1, 2, 3, and 4 numerical representation in this work to represent ASIL A, ASIL B, ASIL C, and ASIL C, respectively. The same as the other categories of the impact (i.e., financial, operational, and privacy), we used the numerical values to represent these categories' different levels from the lowest as "1" to the highest level "4".

5. **Likelihood:** is the probability of an event occurs. The evaluation of the likelihood considers the significant factor for the risk assessment process. While for the transportation area it is addressed to consider the impact only in the risk evaluation [Bra14]. Details of the

Table 6.7: Impact levels

Factor	Impact Levels			
Safety	**ASIL A (1)**	**ASIL B (2)**	**ASIL C (3)**	**ASIL D (4)**
Financial	1	2	3	4
Operational	1	2	3	4
Privacy	1	2	3	4

Table 6.8: Proposed factors of the likelihood parameters

Factors	**Values**			
Capabilities	Amateur **(4)**	Mechanic, Repair shop, etc. **(3)**	Hacker, Automotive expert, etc. **(2)**	Expert, etc. **(1)**
Availability	public **(4)**	information for maintenance availability **(3)**	Information for OEM, system integrator, etc. availability **(2)**	information for ECUs' company availability **(1)**
Reachability	untrusted network **(4)**	private network **(3)**	part time accessible **(2)**	physical access **(1)**
Equipment needed	standard devices **(4)**	Specialize devices **(3)**	Tailor-made device **(2)**	Multiple Tailor-made devices **(1)**

likelihood assessment are presented in [SMR+16], but in short, we propose to evaluate the likelihood from four different factors [SMR+16]:

- assumed attacker capabilities,
- ease of gaining information about the systems,
- reachability and accessibility of the system,
- required equipment for an attack.

These values are defined as scores representing likelihood parameters to estimate the overall likelihood's factor (i.e., Capabilities, Availability, Reachability, and Equipment needed). For example, if an attacker is an expert (i.e., score = "1"), the attack probability is 100% than if the attacker's capability is an amateur (i.e., attack probability = 25%). Equation 6.1 gives a general formula for estimating the probability of each score in Table 6.8.

$$\textbf{Score\_Probability} = 100\% - (Attack\_Score - 1) * 25\% \tag{6.1}$$

where:

- Score_Probability: the probability of likelihood's factors.
- Attack_Score: attack scores based on values in Table 6.8.

Table 6.9 gives an estimation of attack scores and its probability according to the proposed values in Table 6.8.

Table 6.9: Probability of the likelihood attack scores

Attack_Score	Score_Probability
1	100%
2	75%
3	50%
4	12%

These scores are used in the risk assessment process to evaluate each factor and then estimate the overall likelihood parameter.

6. **Safeguards:** once system problems exist, the analyst shall start to select the proper safeguards. The process includes identifying restrictions on the safeguards, of identifying selected safeguards, evaluating, and ranking the safeguard results to find the best ones [CVWF].

According to the ISO 275005 based on the ISO 31000, risk analysis methodologies are classified into three types, such as qualitative or quantitative, or a combination of these two methodologies [iso11].

Qualitative risk analysis: it uses a scale of attributes to describe the value of the impact (consequences) and the likelihood, which is the probability of a consequence that could happen [iso11]. The qualitative risk approach is quicker and simpler to be implemented. This approach focuses on identifying the likelihood and impact of a particular risk during a specific task's life cycle. Depending on these values, the risk severity level is calculated to determine the main objective of this task [bRW19].

Quantitative risk analysis: it uses a scale with numerical values of the impact and likelihood of data from different resources. This process depends on the fullness of impact and likelihood numerical values. The research uses historical incident-based data, which has a benefit if it can be based on security objectives. Nonetheless, it has a drawback, which is the lack of data or new risks or security vulnerabilities information[iso11]. This analysis is considered an objective approach, which uses data to analyze the effects of risk [bRW19].

The proposed risk analysis approach in this work is considered a qualitative methodology because our approach aims to define risks that have impacts against vehicular units, and then measure relevant values of the likelihood and impact to define the risk severity for each identified threat. The likelihood and impact evaluation will be discussed in Section 6.4.3. This evaluation process is essential in estimating the main target level of vehicular units, and in selecting the applicable security requirements for addressing risks and protecting vehicular units from different malicious actions.

A simple sketch design of the risk analysis process is described in Figure 6.17, the threat analysis process aims to identify threats that have an impact against vehicular assets/components according to the existence of security vulnerabilities. The analyzer consists of two primary operations, the

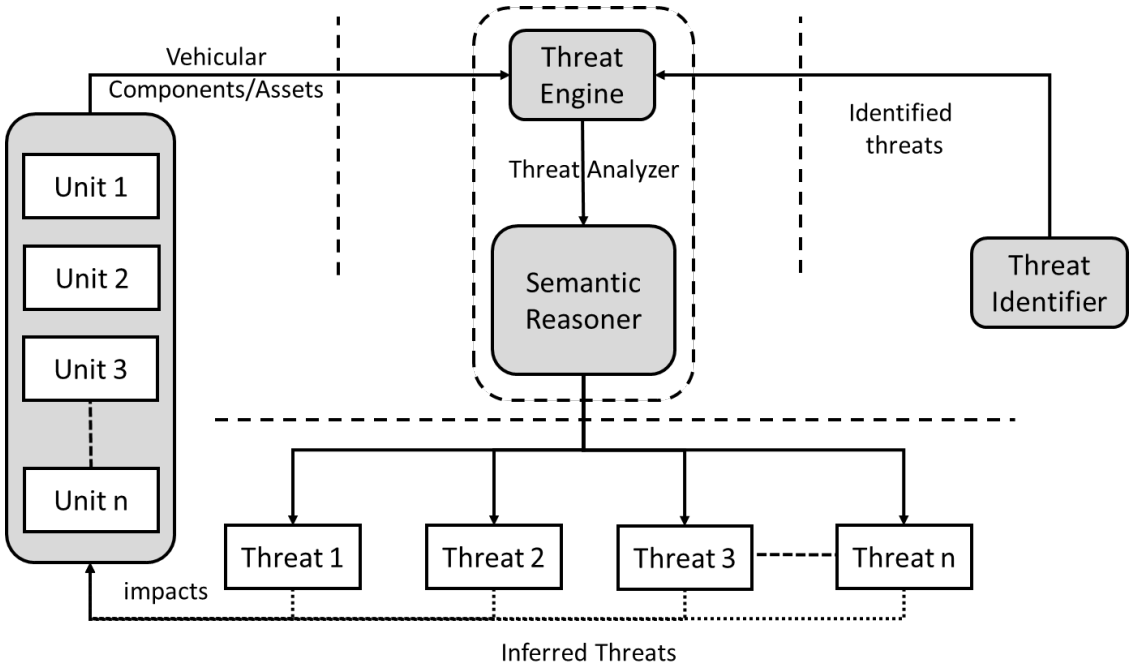


Figure 6.17: The structure of the risk analysis process

threat engine and the semantic seasoning. The threat engine scans all the incoming threat rules obtained from the risk identification process, and gets more information about the selected threats from the threats KB. This information (i.e., likelihood, impact, and category) will be combined within the rule, then processed by the reasoner. The semantic reasoner infers a set of logical consequences from the input to generate a set of potential threats that have an impact against the selected vehicular component/asset. The following algorithm shows the main steps of the rule engine functionality.

Inferred threats are collected as a set of potential threats that have an impact against vehicular units. Then the framework applies the risk assessment strategy for estimating severity degree for each an identified threat to define the risk level. The risk level helps map risk degree with appropriate security levels of security requirements for addressing these risks.

6.4.3 Risk Evaluation

The next process evaluates the risk for each threat identified, to calculate a severity level's preliminary estimation. The risk assessment process uses several risk methods for evaluating the vehicular risk level based on the parameter values of the likelihood and impact. The following formula is one of the most common risk assessment methods:

$$Risk = Threat * Vulnerability * Consequence \tag{6.2}$$

where:

$Threat * Vulnerability$ = Likelihood
$Consequence$ = Impact

Data: Infer potential threats that have an effect on vehicular components
Result: Set of potential threats
initialization;
Read Component;
Read ThreatsDB;
Set inferedResult;
Set InferThreats[];
while !isEmpty(ThreatsDB) **do**
  Set queryEngine; "SQWRL Query Engine";
  inferedResult = SemanticReasoner(queryEngine, Component, ThreatsDB);
  while !isEmpty(ThreatsDB) **do**
    if !InferThreats.contains(inferedResult) **then**
      InferThreats.add(inferedResult);
    end
  end
end

Algorithm 6: Infer a set of threats that affect a particular component

Table 6.10: An example of risk matrix

		Impact			
		1 Minor	**2 Moderate**	**3 Major**	**4 Critical**
Likelihood	**1 Unlikely**	1	2	3	4
	2 Possible	2	4	6	8
	3 Likely	3	6	9	12
	4 Certain	4	8	12	16

In this work, we build the risk table based on four levels of risk severities, because we also classify security requirements into four security levels. Therefore, compatibility between risk levels and security requirements is needed to allow the matching process between these parameters. Table 6.10 is an example of a risk matrix as explained in IEC 62443-3-2 [IEC15].

In order to estimate the likelihood and impact parameter values of threats incorporated in this work, we use the proposed values in threats list by the UNECE [UNE17b]. Also, we update these values according to personal knowledge and individual endeavours based on discussions with experts from AIT-Austrian Institute of Technology [5] in the vehicular domain to identify set approximate numerical values of risk assessment values to help in mitigation existing risks. A screenshot of the UNECE threats list is illustrated in Figure 6.18.

The last two sections in the UNECE's threats list explain how an attack could be initiated and describe the expected consequence for each attack. We apdated parameters in these sections to help in the likelihood and impact estimation processes.The next sections discuss methods followed

[5] https://www.ait.ac.at

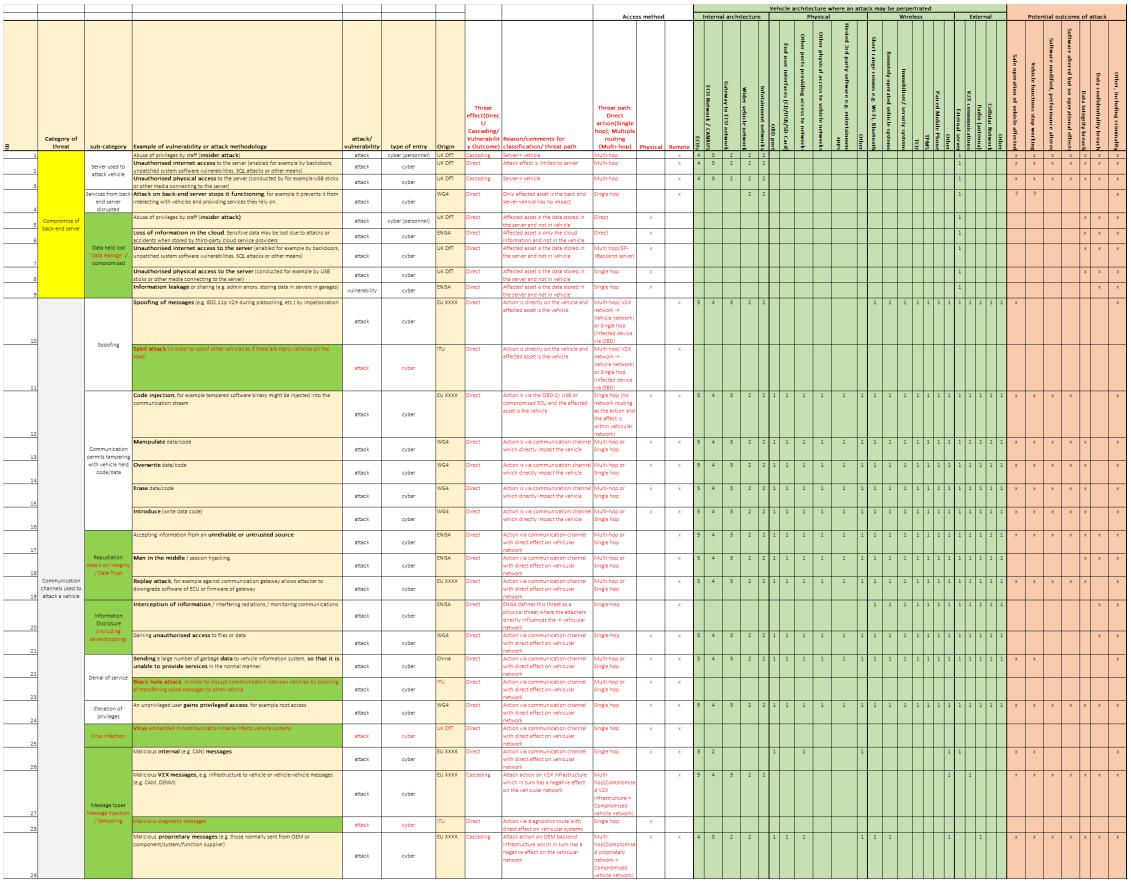


Figure 6.18: A screenshot of the UNECE threats list, as described in [UNE17b]

in this work to evaluate values of the likelihood and the impact.

Estimation of the Likelihood Parameter Values

As discussed earlier, the UNECE threats list describes how different vehicle units might be the way for an attack to be triggered. In this work, we use these descriptions and improve them to define a set of likelihood values of an attack based on a specific vehicle unit, which an attack could be initiated from. The estimation process of the likelihood is based on four factors as discussed and presented in Table 6.8, as descibed in [SMR+16]. Furthermore, the likelihood value for a particular threat is evaluated according to the calculation value of these four categories. According to UNECE [UNE17b], the second last group of columns focuses on the categories and sub-categories of vehicular units that an attack could be initiated from, as shown in Figure 6.19.

During this assessment, the proposed parameter values in Table 6.8 are also used to estimate the probability from various perspectives (i.e., capability, availability, reachability, and equipment needed). For example, physical and internal architecture categories that take the highest data reachability value (e.g., 1), as described in Table 6.9. Since these two categories are introduced when the attacker is directly connected to the vehicle network. On the other hand, the other two forms (i.e. wireless and external) may take 3 to 4 values as the reachability level is lower than a physical system connection. These proposed values are described as multiplication factors in the

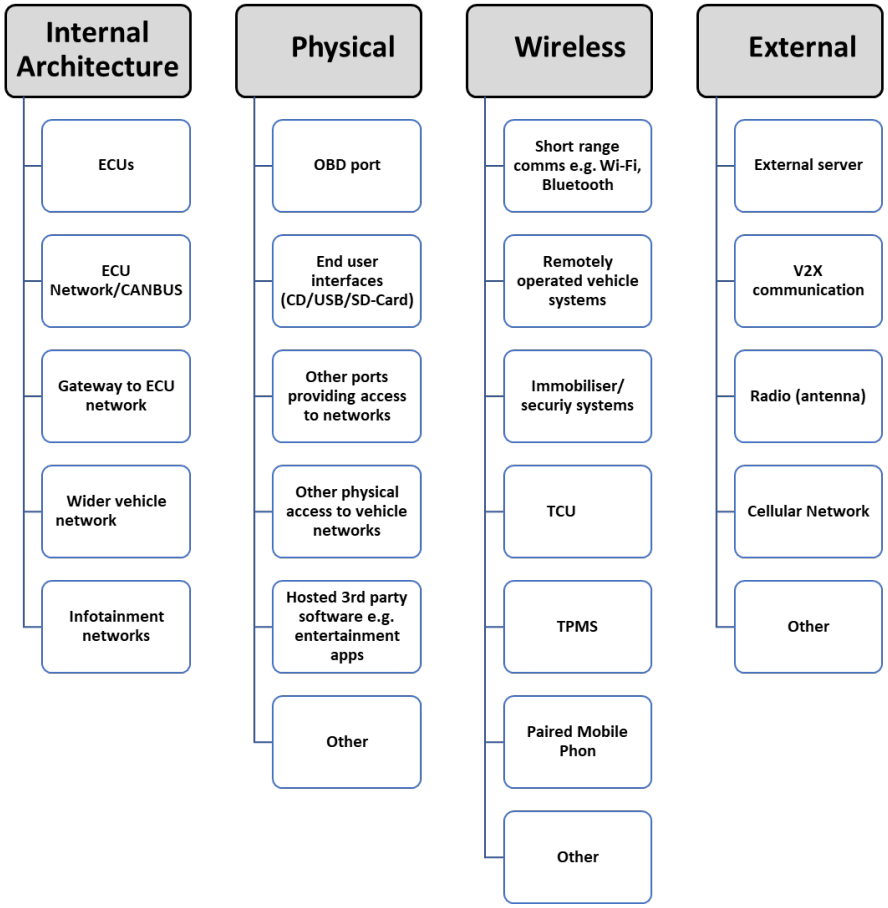


Figure 6.19: Vehicle architecture components where an attack may be initiated from

range from 1 to 4 to empower the estimation of particular probability perspectives than others. In order to determine the reachability factor, Equation 6.3 is applied.

$$\textbf{Reachability'} = \frac{1 * (\sum Physical + \sum Internal) + 4 * \sum Wireless + 3 * \sum External}{Max(Reachability)} * 100 \quad (6.3)$$

where:

- Max(Reachability): define the maximum value of reachability for vehicular assets.
- Reachability': define the percentage estimation of the reachability factor.

The capability factor is based on the ability of an attacker to perform a malicious action. According to Table 6.8, four scores define the capability of attackers. Regarding the complexity of the vehicular structure, which needs high skills to perform malicious actions against the vehicular network. In this work, we consider the capability to attack the vehicle could be classified as "hacker" or "expert", which takes factors between 1 to 2, as the highest probabilities for a successful attack, as described in Tablefactorscore. Equation 6.4 is applied to measure the

estimation value of the capability factor.

$$\textbf{Capability'} = \frac{1 * (\sum Physical + \sum Wireless + \sum External) + 2 * \sum Internal}{Max(Capability)} * 100 \tag{6.4}$$

where:

- Max(Capability): define the maximum value of attacker capabilities.
- Capability': define the percentage calculation of the capability factor.

Based on the level of asset availability, the direct relationship between an attacker and an asset itself may be established. For example, in-vehicle internal ports like an On-Board Diagnostics (OBD) port may be a direct way for an attacker to perform alternative malicious behaviour against vehicle systems when the attack is classified as physical access. Equation 6.5 is used in order to estimate the value of asset availability.

$$\textbf{Availability'} = \frac{1 * (\sum Internal) + 2 * \sum Ports}{Max(Availability)]} * 100 \tag{6.5}$$

where:

Ports:

- Availability': represent the percentage value of an asset availability.
- Max(Availability): the maximum value that represents the availability of an asset.

Additionally, in the likelihood estimation, the equipment required is often used to determine the degree of equipment used by attackers to attack a particular asset or the entire vehicular network. For this assessment, tailor-made or multiple tailor-made devices are required to assist attackers in performing various attack actions. Thus, according to the values of the equipment needed, as described in Table 6.8, scores 1 and 3 are used to calculate the value of the equipment needed in the likelihood calculation process. Equation 6.6 is applied to evaluate the value of the equipment needed in the attack process.

$$\textbf{Equipment'} = \frac{1 * (\sum Internal + \sum Physical) + 3 * (\sum Wireless + \sum External)}{Max(Equipment)} * 100 \tag{6.6}$$

where:

- Equipment': define the maximum representation value of the tailored equipment is used to attack a particular asset.
- Max(Equipment): the maximum value of that represent the degree of the equipment are needed to attack a particular asset within the vehicular system.

The Equation 6.7 is applied to evaluate the likelihood value of each threat. In order to evaluate the overall risk, the percentage of the probability should be adjusted to match the Equation. 6.2.

$$\textbf{Likelihood'} = \frac{(Capability' + Availability' + Reachability' + Equipment')}{4}\% \tag{6.7}$$

Therefore, we recalculate the percentage of the estimated probability into a decimal value between the range from 1 to 4 to fully adapted with the risk matrix as described in Table 6.10. Equation 6.8 is applied for each likelihood percentage value to obtain the required value.

$$\textbf{Likelihood} = \lceil \text{Likelihood}' * 25\% \rceil \tag{6.8}$$

The likelihood evaluation of some selected potential threats is depicted in Figure 6.20.

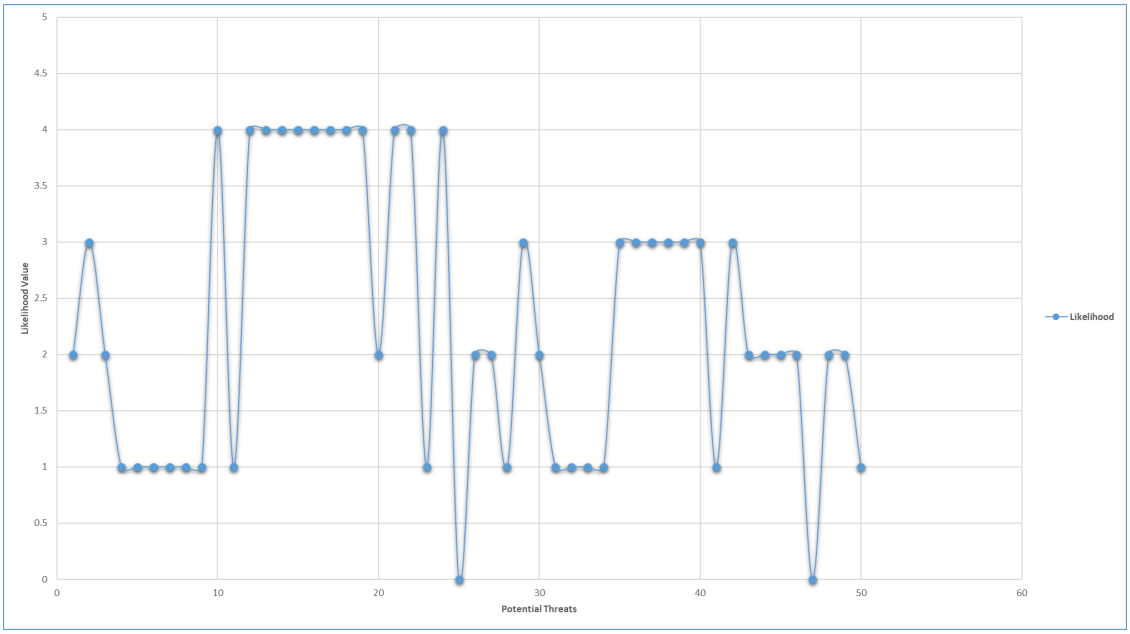


Figure 6.20: The likelihood evaluation of some selected threats

The above figure shows the evaluation of likelihood parameters of some selected potential threats from the UNECE list. The evaluated values are defined according to the previously discussed formulas.

Estimation of the Impact Parameter Values

Seven impact criteria identify the potential impact in UNECE list [UNE17b]. These criteria are defined as safe operation, vehicle functions stop working, software modified, performance altered, software altered, but no operational effects, data integrity breach, data confidentiality breach, and other including criminality. In this work, we study these criteria for each threat based on personal diligence to estimate a fundamental impact level. The impact estimation for each threat is defined according to the proposed impact parameters, as discussed in Table 6.7. The overall impact value for each threat is calculated according to the impact categories (i.e., safety, financial, operational, and privacy). Some identified potential impact parameters from the UNECE list could be selected to calculate the different impact factors (i.e., safety, financial, operational, and privacy). For example, the safety parameter is evaluated according to multiple values of a potential impact, such as:

- Safe operation of vehicle affected
- Vehicle functions stop working

Table 6.11: The numerical range of threat outcomes

Potential outcome of attack	Numerical Range of the Impact
Safe operation of vehicle affected	1 - 4
Vehicle functions stop working	1 - 4
Software modified, performance altered	1 - 4
Software altered but no operational effects	1 - 4
Data integrity breach	1 - 4
Data confidentiality breach	1 - 4
Other, including criminality	1 - 4

- Software modified, performance altered
- Other, including criminality

Each of the above parameters is multiplied by various factors (f) from 1 to 4 as defined in Table 6.11, reflecting the impact of each attack on vehicle systems.

These numerical rages are described according to some estimation values for the outcome for each attack. The following Equation 6.9 is applied to estimate the parameter value of the safety factor;

$$Safety' = \frac{(sf1 * S + sf2 * V + sf3 * SM + sf4 * O)}{sf} \tag{6.9}$$

where:

- **S**: Safe operation of vehicle affected
- **V**: Vehicle functions stop working
- **SM**: Software modified, performance altered
- **O**: Other, including criminality
- **sfn**: is the safety factor.
- **sf**: sum all selected safety factors for estimating the safety level of a particular threat.

The second evaluation parameter is the financial factor, representing the financial damage or consequences when a particular attack happens. To evaluate the value, we check which of the proposed potential outcome parameter could affect this factor. There are two parameters that could affect the financial consequence, defined as follows:

- Data confidentiality breach
- Other, including criminality

Equation 6.10 is applied on the selected affected parameter to estimate the value of the financial parameter.

$$Financial' = \frac{(ff1 * Con + ff2 * O)}{ff} \tag{6.10}$$

- **Con**: Data confidentiality breach
- **O**: Other, including criminality
- **ffn**: the financial factor.
- **ff**: sum all the selected financial factors to estimate the financial impact level for a particular threat.

The operational factor is evaluated according to the proposed values of software modified, performance altered, vehicle functions stop working, and other, including criminality. Therefore, the following Equation 6.11 is applied in order to evaluate the value of the operational parameter.

$$Operational' = \frac{(of1 * V + of2 * SM + of3 * O)}{of} \tag{6.11}$$

- **V**: Vehicle functions stop working
- **SM**: Software modified, performance altered
- **O**: Other, including criminality
- **ofn**: the operational factor.
- **of**: sum all the selected operational factors to estimate the operational impact level for a particular threat.

The last factor in the impact evaluation process is to estimate the consequence of an attack on the privacy; hence Equation 6.12 is used here in order to calculate the value of the privacy factor.

$$Privacy' = \frac{(pf1 * DI + pf2 * Con + pf3 * O)}{pf} \tag{6.12}$$

- **DI**: Data integrity breach.
- **Con**: Data confidentiality breach.
- **O**: Other, including criminality.
- **pfn**: the privacy factor.
- **pf**: sum all the selected privacy factors to estimate the privacy impact level for a particular threat. Equation 6.9.

The overall impact value of each threat is evaluated according to the following Equation 6.16.

$$Impact = \frac{(Safety' + Financial' + Operational' + Privacy')}{4} \tag{6.13}$$

Figure 6.21 illustrates the evaluation of the impact values of each threat, according to the Equation 6.16.

The figure shows the evaluation of impact parameters of some selected potential threats from the UNECE list. The evaluated values are defined according to the previously discussed formulas.

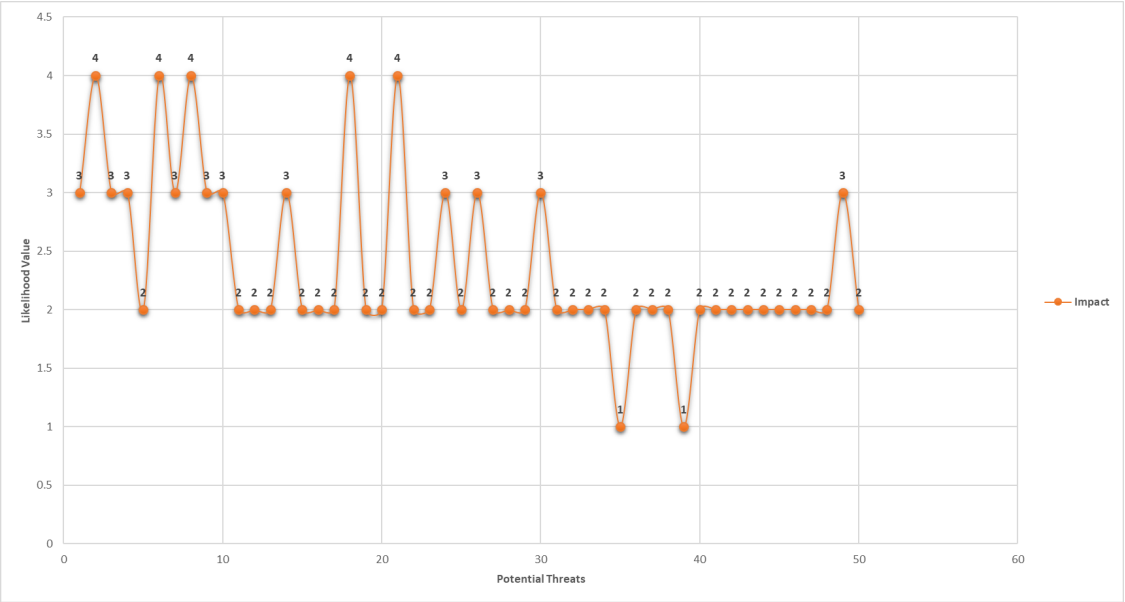


Figure 6.21: The impact evaluation of some selected threats

6.4.4 Risk Evaluation

We use the obtained results from the likelihood and impact to evaluate the overall risk for each threat individually. Equation 6.2 is used as one of the most common risk assessment methods to estimate each threat's risk severity level. The risk estimation value is essential to define the main security target (ST) for a particular component. Therefore, the security target of a specific component/asset is determined according to the max risk level of all impacted potential threats against this component/asset. Each risk level has a determined severity level, which represents the violation degree in case particular threat happens. In order to estimate this degree, we use the risk mitigation process based on the Cyber Risk Reduction Factor (CRRF) as described as in [IEC15]. CRRF measures the degree of security risk reduction required to achieve tolerable risk. The equation 6.14 presents the calculation formula of CRRF.

$$CRRF = \frac{Risk}{TolerableValue} \tag{6.14}$$

Table 6.12 shows the numerical values of the risk levels based on the TV equal 3, as described in [IEC15].

Figure 6.22 depicts the results of the evaluation process of the risk assessment process.

The figure displays the risk estimation values of the selected based on the evaluation of the likelihood and impacts as depicted in Figure 6.20 and 6.21. The coloured zones in the figure represent the degree of severities of these threats according to the estimation parameters in Table 6.12.

6.4.5 Risk Treatment

Risk treatment is a risk control process is focused on handling the identifying risk by proper security requirements according to the evaluation process, as discussed in Section 6.4.3. The risk

Table 6.12: The risk levels according to the CRRF, when the tolerable value = 3

Level	Risk	CRRF	Severity Level
Low	1	0.33	1
	2	0.67	1
	3	1	1
Medium	4	1.33	2
	5	1.67	2
	6	2	2
High	7	2.33	3
	8	2.67	3
	9	3	3
Critical	10	3.33	4
	11	3.67	4
	12	4	4
	13	4.33	4
	14	4.67	4
	15	5	4
	16	5.33	4

treatment actions can be divided into four categories (i.e., mitigation, avoidance, acceptance, or transfer).

Risk Avoidance: if the identified risk has a high impact and likelihood, it is essential to address it by implementing security measures. These measures manage and handle this issue and avoid any negative consequences that could affect the vehicular system.

Risk Mitigation: is another action that could be considered against a particular risk when the risk has a high impact on the vehicular units or the whole system. However, when a risk with a high impact value with a low probability of occurrence, it needs to be handled by implementing security measures to reduce the risk's consequences.

Risk Acceptance: in this case, no particular action is required because the risk has low impact and likelihood, which could be accepted.

Risk Transfer: this type of action treats risks when having a high impact on the vehicular systems and low probability. Therefore, these risks could be transferred to someone or an organization that can handle and manage risks.

It is also discussed in ISO/IEC 27005 [iso11]; the risk treatment consists of four options (e.g., modification, retention, avoidance, or sharing). The risk treatment process plays a vital role in the risk management process for addressing identified threats to be tolerable and acceptable. A secure vehicle can be implemented and designed only if:

- the potential threats are identified,
- risk severity levels are correctly estimated

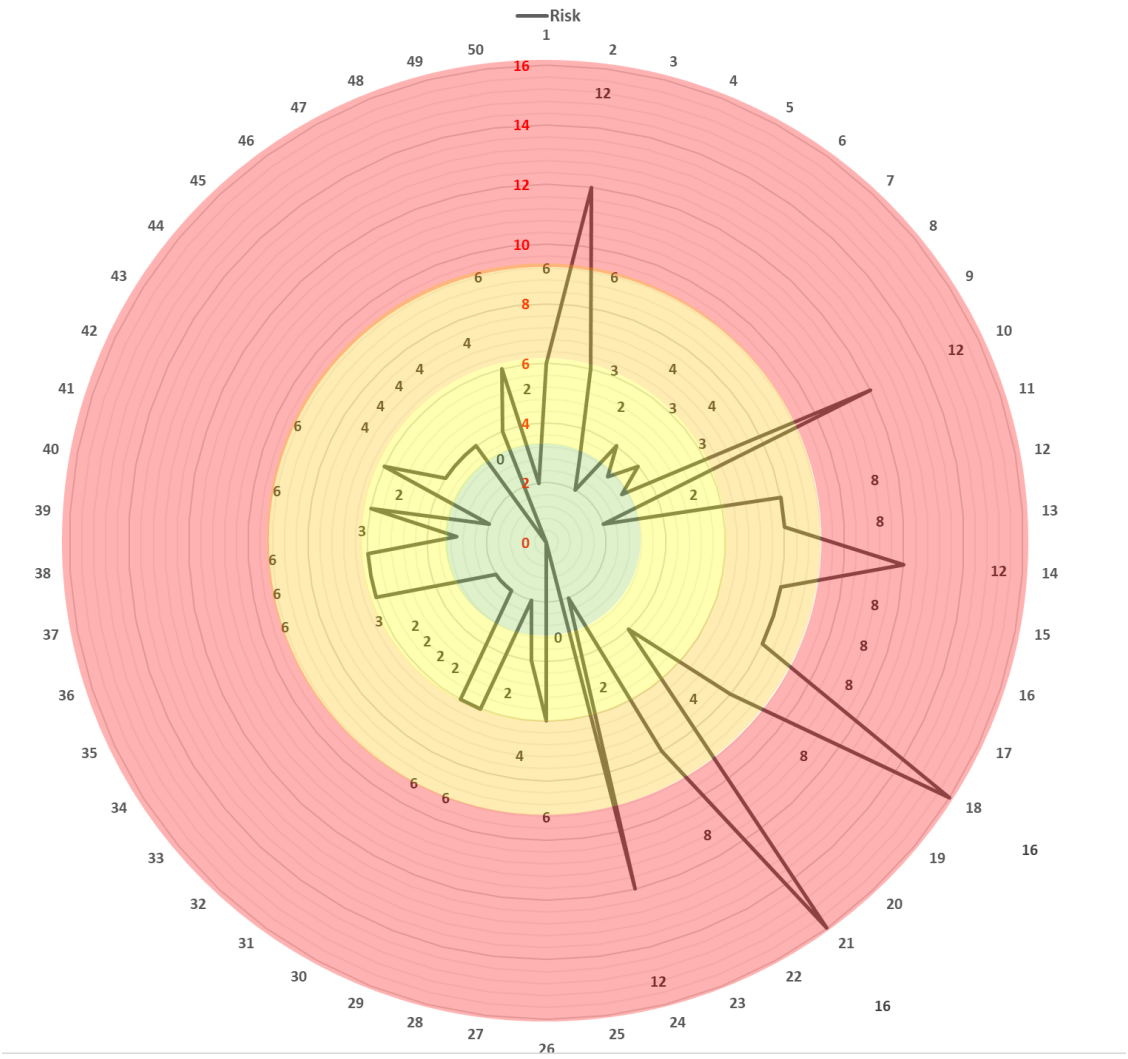


Figure 6.22: The risk evaluation results of some selected potential threats

- security measures are properly selected

Therefore, in this thesis, we aim to keep the risk always low; hence, the risk mitigation and avoidance actions are used to achieve this task. In addition, the risk acceptance is also included in this work, when a risk has low impact and likelihood parameters, or also if proper security measures already address that risk. The results are defined as a set of identified threats, and relevant risk severity levels define each identified threat's real severity consequence. For example, we expect that a risk level of a threat is evaluated using equation 6.2; this risk is estimated as a high impact on a specific vehicular unit. The estimated risk is illustrated at point (T1) in Figure 6.23. In order to perform a correct risk evaluation process, we define as a risk threshold tolerable risk value (TV), which means all values above this threshold (TV), need to be addressed by the suitable security countermeasures to mitigate this risk. In this example, we claim that the TV is equal to 3.

T1 on the figure shows that this threat is classified as critical after the risk evaluation process. Therefore, the security countermeasure(s) are used to mitigate this risk. This state is defined as the Security Achieved (SA), which aims to decrease the likelihood of the unaccepted risks.

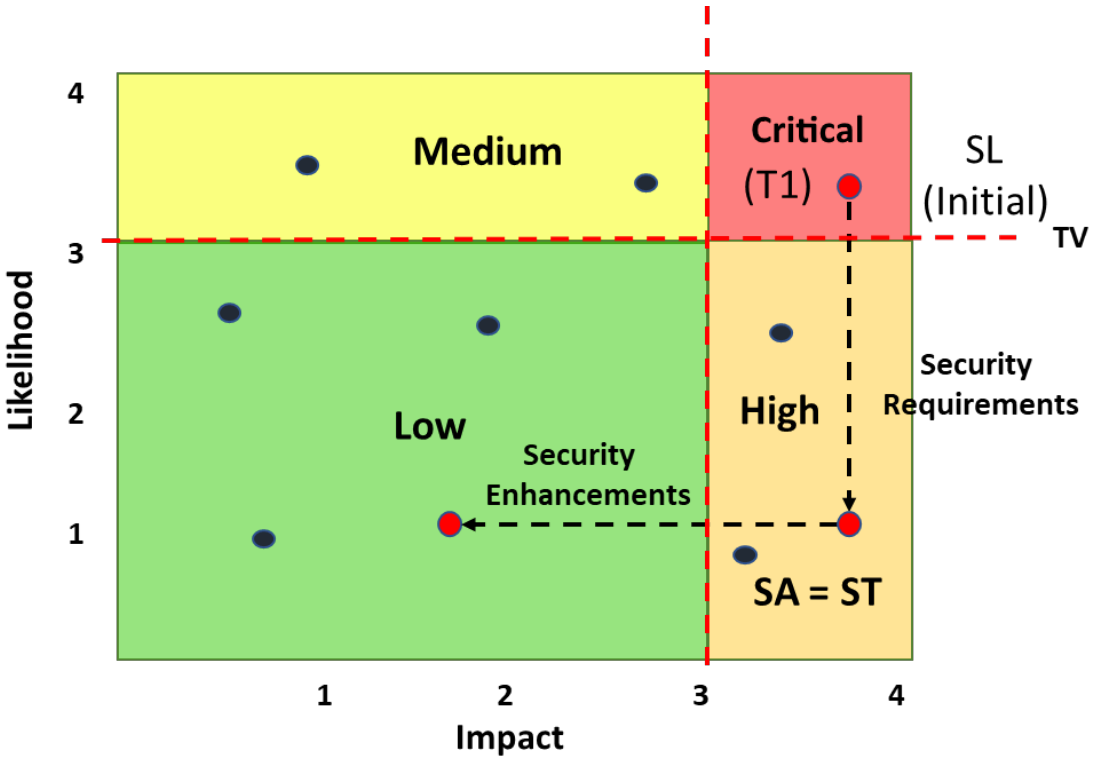


Figure 6.23: The risk treatment sequence for mitigating a particular risk

Then confirms that is security achieved meets the actual security needed "security target (ST)" by checking SA==ST. If not, other security requirements are used to manage this risk and mitigate it into an acceptable one. A sequence of processes is specified to address a particular risk by a proper set of security requirements to fill the exiting security gap. Figure 6.24 illustrates the main steps in the risk treatment process.

As described in the figure, the identified threats, exploited security properties/measures, and affected units are considered the main data input in this process. These inputs are processed to select the relevant security requirements that can treat each threat's estimated risk. The "Security Rule Builder" function is responsible for generating semantic rules according to the input data. The vehicular specifier classifies the vehicular component class according to the security requirements classes, as is described in IEC 62443-4-2 [IEC19]. The IEC 62443 part 4-2, provides component requirements and classifies these requirements into four groups, as defined as follows [IEC19]:

- Software application requirements (SAR),
- Host device requirements (HDR),
- Embedded device requirements (EDR),
- Network device requirements (NDR)

The components are classified into one of these groups according to each component's functionality in the vehicle. For example, an electronic control unit (ECU) in the vehicular embedded unit performs processing of a set of data input from alternative sources to deliver a particular action to

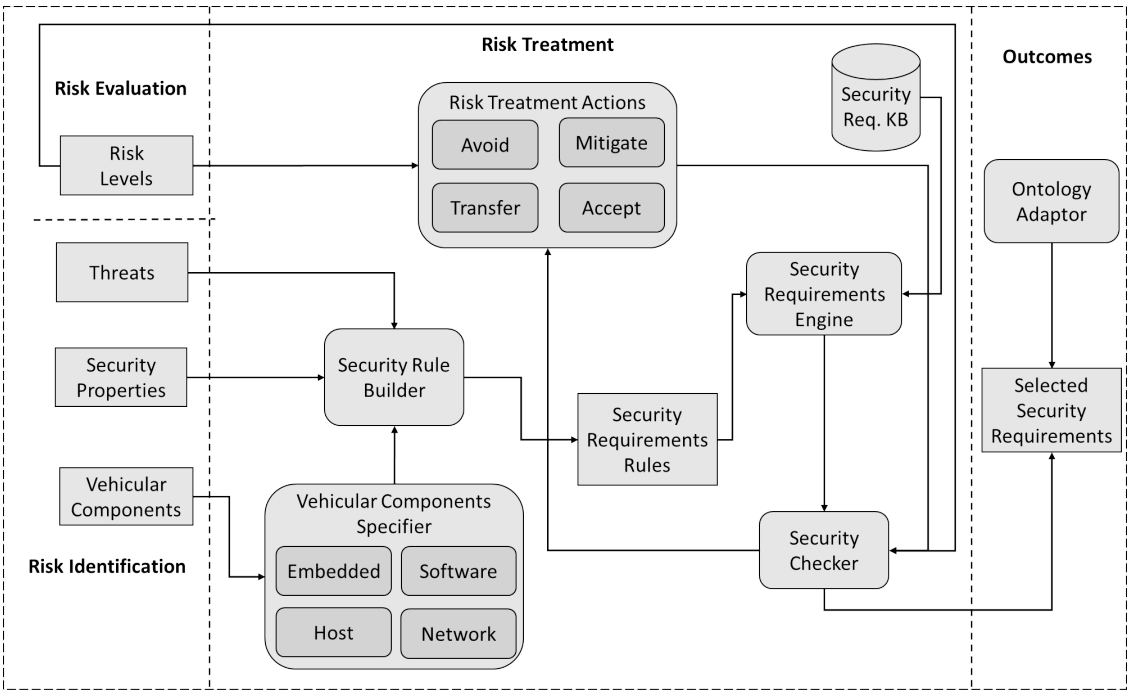


Figure 6.24: The building blocks of the security requirements process

a next-hop unit such as an actuator. Therefore, the unit is classified as an embedded device. The risk treatment process intends to explore the complete set of security requirements that meet the vehicular unit's corresponding security measures. Also, meet the same class type of vehicle unit to choose the most suitable solution that manipulates the current risk. In some cases, in the security requirements selection process, no matched contents are identified. Therefore, the risk treatment process selects the most relevant security requirements matching security characteristics with the vehicular elements' security measures. The search technique applies a set of inference rules that can select a group of security requirements from the general collection of security requirements as defined in the IEC 62443-4-2 as a Component Requirement (CR) [IEC19]. The selection of security requirements is based on the security measures of a particular component/asset, which threat(s) can breach to exploit the existing security vulnerabilities. Therefore, the security measures are the key to a successful choice of security requirements, as is illustrated in Figure 6.25.

Security measures are the key point in determining acceptable security requirements that need to be part of vehicle components/assets to protect against cyber risks. As shown in the figure, the three relationships (i.e., hasProperts, hasMitigation, and exploits) among ontology individuals should be clearly defined to infer specific security requirements to fill security gaps. The Security Builder function is responsible for collecting all relevant data to establish logical rules for inferring security requirements. Algorithm 7 explains the process of developing semantic rules for selecting security requirements based on the relationship between security measures, threats, and components/assets, as previously identified.

An example of the generated semantic rule is defined in Liting 6.6:

```
1 OnSecta1:Components(OnSecta1:VCS_data) ^ OnSecta1:classifiedAs(OnSecta1:
    VCS_data, OnSecta1:pp) ^ OnSecta1:Threats(OnSecta1:T13_6) ^ OnSecta1:
    Security_Requirements(?SR) ^ OnSecta1:classifiedAs(?SR, OnSecta1:pp) ^
```

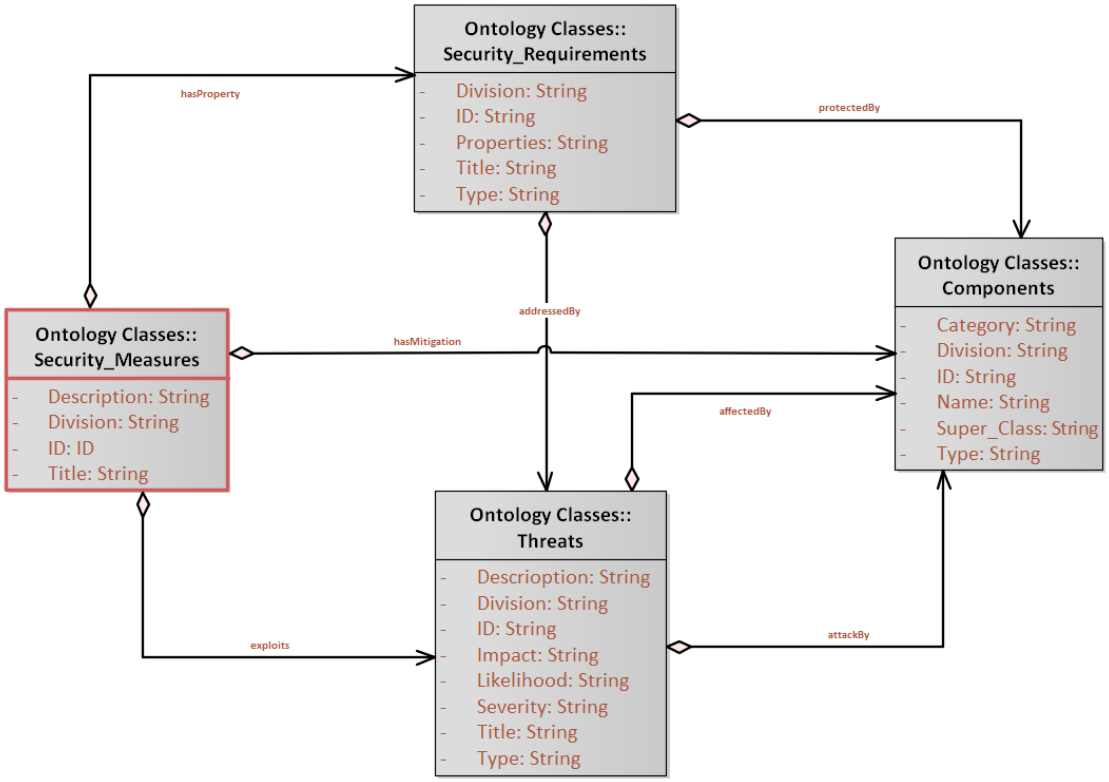


Figure 6.25: The relationship of the security measures with security requirements, components, and threats

```
    OnSecta1:hasMitigation(OnSecta1:VCS_data, OnSecta1:integrity) ^
   OnSecta1:exploits(OnSecta1:T13_6, OnSecta1:integrity) ^ OnSecta1:
   hasProperty(?SR, OnSecta1:integrity) -> OnSecta1:impacts(OnSecta1:
   T13_6, OnSecta1:VCS_data) ^ OnSecta1:protects(?SR, OnSecta1:VCS_data)
^ OnSecta1:addressedBy(OnSecta1:T13_6, ?SR)
```

Listing 6.6: Sample of a generated semantic rule for selecting applicable security requirements

This rule aims to select security requirements that can fill security gaps in vehicle design, particularly in components/assets. After applying the created rules, new relationships will be identified and integrated with the vehicle taxonomy. The taxonomy structure assists in outlining a complete map of the vehicle system with all the relevant details (i.e., threats, security measures, components, assets, security requirements, and risks). This helps to observe existing security vulnerabilities in vehicle design (as problems) and identify the relevant security requirements (as solutions) that can overcome existing security issues.Figure 6.26 depicts a sample of the vehicle ontology taxonomy before applying the previously defined semantic rule.

The V2X_HSM is a security module component, as discussed in [Car19]. The T13_6 violates the integrity of VCS data; therefore, integrity is considered a security gap in vehicle design, particularly in the VCS_data asset unit. Therefore, it is important to define proper security requirements, as necessary security measures that need to be part of the VCS data asset to achieve a specific security objective. A single vulnerable point could have different risk consequences

Data: Creating semantic rules for selecting security requirements
Result: Set of security requirements for filling existing security gaps in the vehicle components/assets
initialization;
Read the ontology_hierarchy;
Read Security_Measure;
Read ComponentName;
Read ComponentType;
Read ThreatID;
Read exploitName;
Set SR; "Security requirement" Set Rule_Set[]; "A list of generated rules" Rule_Structure;
while !isEmpty(AllExploitations) **do**
  $Rule\_Structure =$
  $Components(ComponentName) \wedge classifiedAs(ComponentName, ComponentType) \wedge$
  $Threats(ThreatID) \wedge Security\_Requirements(?SR) \wedge$
  $classifiedAs(?SR, ComponentType) \wedge exploits(ThreatID, Security\_Measure) \wedge$
  $hasProperty(?SR, exploitName) -> impacts(ThreatID, ComponentType) \wedge$
  $protects(?SR, ComponentName) \wedge addressedBy(ThreatID, ?SR)$
  Rule_Set(Rule_Structure);
end

Algorithm 7: Create security requirement rules

on either a single point in the vehicle , or it could be a point in a full attack path that includes one or multiple weak points that an attacker follows to achieve a malicious goal. Therefore, this single weak point should be secured in both situations, and the current gap must be addressed to achieve the security objective. Therefore, the process of analyzing security requirements plays an important role in determining the best security requirement to solve existing security issues. The SWRLAPI [6] is used as a security requirement engine to support semantic reasoning. The engine uses the set of created rules and applies the semantic reasoning method to infer a new set of security requirements stored in the security requirements knowledge-base. The newly selected security requirements are then incorporated with the ontology model to extend the vehicle taxonomy with all relevant security information. Figure 6.27 depicts the new form of the taxonomy after the rule is applied, and the rule engine prioritizes a new set of security requirements.

The relationships "impacts," "protects," and "addressedBy" are identified and incorporated within the vehicle taxonomy, as is depicted in Figure 6.27. These relationships are described as:

- **Impacts:** the relationship represents a direct connection from a threat to affected component/asset.
- **protects:** the security requirement(s) are selected to fulfill the prerequisite security mechanisms needed to protect a vehicular component/asset. Therefore, the "protects" relationship is defined to represent the relationship between a component/asset and a selected security requirement.
- **addressedBy:** security requirements are chosen to address a particular risk; this relationship is then defined between the security requirement and a particular threat.

[6] https:/github.com/protegeproject/swrlapi

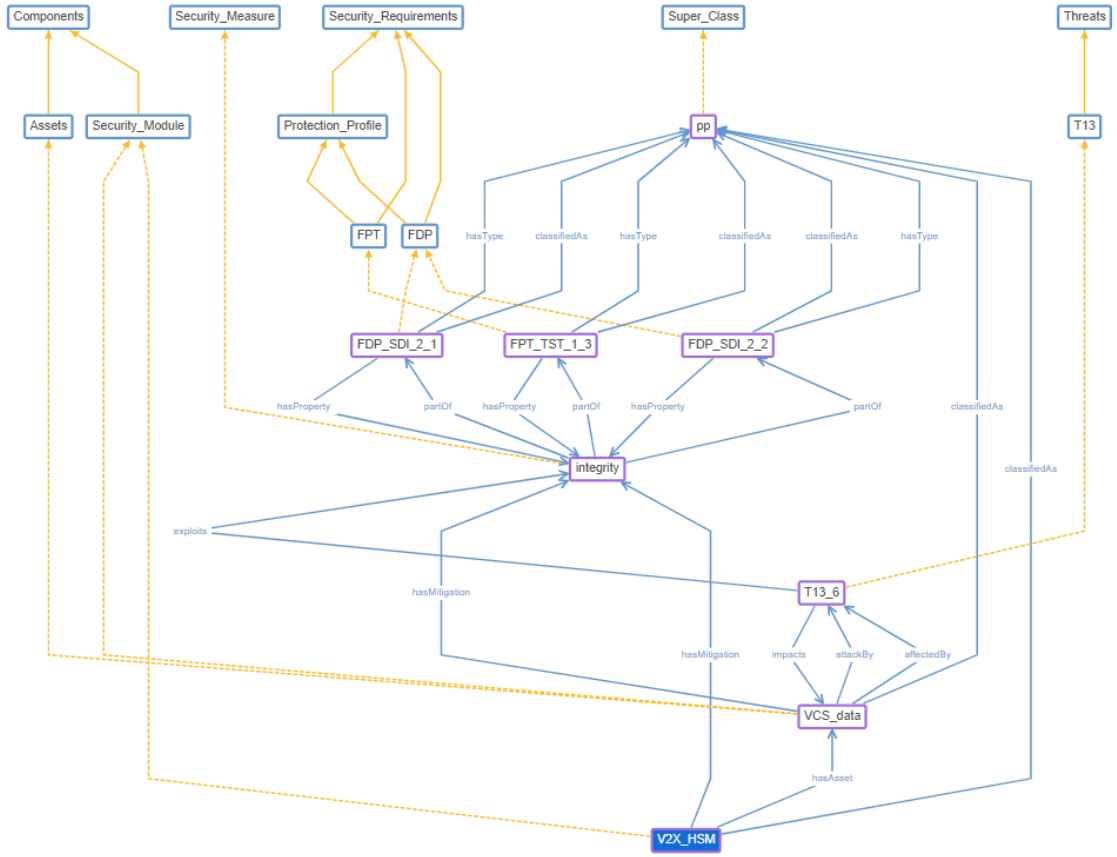


Figure 6.26: The ontology taxonomy of part of the vehicle model before applying the previously generated semantic rule

6.5 Gap Analysis and Security Enhancement

The unsatisfied results collected from the previously discussed processes are obtained and defined as security gaps in the vehicular system design. These gaps need to be managed and handled to fill the security gaps. The proposed framework aims to classify these gaps and define possible solutions that can address the existing security issues. During the concept phase, the Security Target (ST) must be identified for each vehicular component/asset the security goal. In some cases, the selected security requirements are not sufficient to achieve the security goal. Furthermore, the framework plays a vital role in this process to increase the current security level and add additional security requirements that could help in achieving the required security target by deducing applicable security requirements for addressing the security issues. In this work, we focus on evaluating the security achieved according to the inferred security requirements by the framework; we cannot calculate the security achieved of the security requirements because the framework is built to be fully adaptable for many different security requirements forms. The required data for evaluating the security achieved level is rather than could not be available in many security requirements documents. For example, the IEC 62443 security standard, defines the security requirements into four Security Levels (SLs), as discussed in [isa13]:

- SL1: Prevention of unauthorized data disclosure.

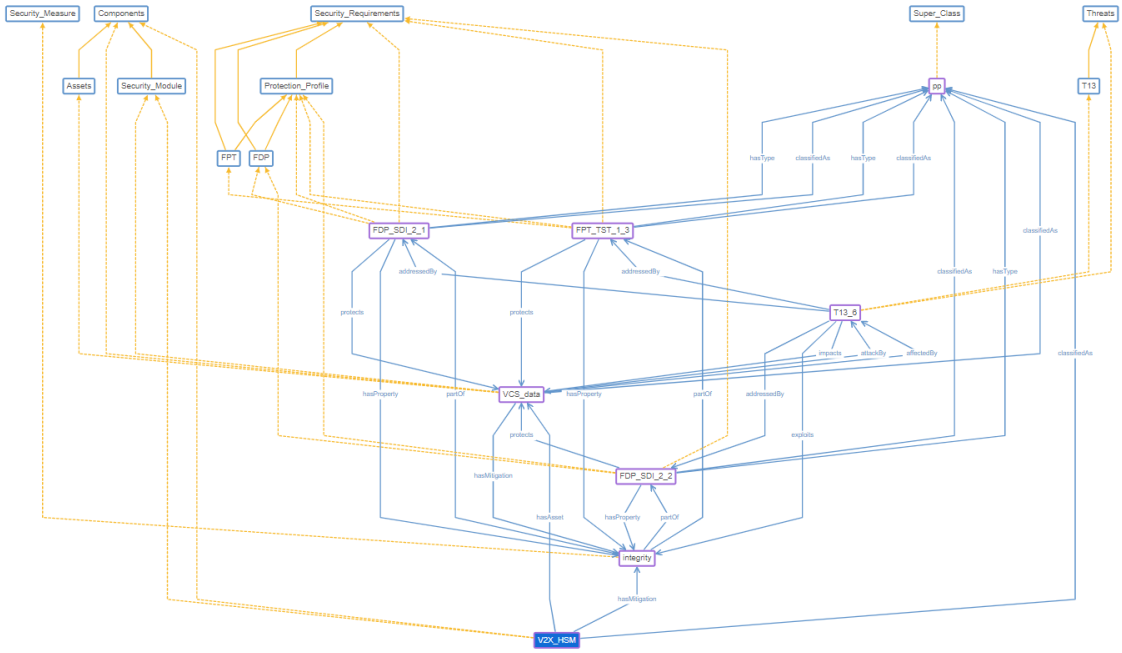


Figure 6.27: The new ontology taxonomy of the vehicle model after the rule is applied

- SL2: Prevention of unauthorized data disclosure with low motivation and general skills.
- SL3: Prevention of unauthorized data disclosure with IACS distinct competencies and moderate motivation.
- SL4: Prevention of unauthorized data disclosure with IACS specific skills and powerful motivation.

We use the values of security levels defined in the IEC 62443 security standard to evaluate an approximate value that represents the security level achieved of a particular set of security requirements for a particular vehicular component/asset. Therefore, evaluating the security achieved of the previously selected security requirements from different resources is out of the scope of this work.

In some examples, vehicle could be implemented according to a set of applied security requirements without care if these requirements are redundant or ineffective, against the exiting security issues. Hence, the framework applies a set of logical operations for checking if the selected security requirements are sufficient to address the expected threats or not, and according to the outcomes of this process, the framework handles these gaps with an appropiate set of security requirements. Then the framework calculates a value that represents the effectiveness of the applied security requirements for protecting a particular vehicle component/asset. This value represents an indication of the security level after applying a set of security requirements. The next section gives an overview of the followed steps to estimate the Security Achieved (SA) level for each component/asset.

6.5.1 Evaluation of the Security Level Achieved

The security level achieved represents an estimated value of the accomplished level after the security requirements are applied to a specific vehicle component/asset. This estimated value is based on security levels for each selected security requirement. The outcome of this process should be equal to the security target level for each vehicular unit. The security target for a particular vehicular component/asset is defined as the max severity level of the impacted potential threats for this vehicular unit. Equation 6.15 is used to select the max severity level (based on the risk evaluation phase, as discussed in Section 6.4.3) of all threats that impact the affected component/asset.

$$\mathrm{Com}_{ST} = \underline{\mathrm{Max}}\,(T1_{ST}, T2_{ST}, \ldots\ldots\ldots, Tn_{ST}) \tag{6.15}$$

The max severity value of threats is selected as the ST that needs to be achieved to protect a particular component/asset. As discussed previously, each threat has a set of security requirements that are selected to mitigate the impact of threats. Therefore, we use the values of the security levels of all these security requirements to estimate a value that represents the effectiveness of these requirements against particular threats. Equation 6.16 applied for each threat to estimate a value that defines the degree of security achievement after a set of security requirements are selected for a particular threat.

$$\begin{aligned}
T1_{SA} &= \frac{\sum_{i=1}^{n}\left(\frac{\mathrm{Min}(SR1_{SL})}{T1_{ST}} * \mathrm{Com}_{ST} + \frac{\mathrm{Min}(SR2_{SL})}{T1_{ST}} * \mathrm{Com}_{ST} + \cdots + \frac{\mathrm{Min}(SRn_{SL})}{T1_{ST}} * \mathrm{Com}_{ST}\right)}{n} \\
T2_{SA} &= \frac{\sum_{i=1}^{n}\left(\frac{\mathrm{Min}(SR1_{SL})}{T2_{ST}} * \mathrm{Com}_{ST} + \frac{\mathrm{Min}(SR2_{SL})}{T2_{ST}} * Com_{ST} + \cdots + \frac{\mathrm{Min}(SRn_{SL})}{T2_{ST}} * Com_{ST}\right)}{n} \\
&\cdot \\
&\cdot \\
&\cdot \\
Tn_{SA} &= \frac{\sum_{i=1}^{n}\left(\frac{\mathrm{Min}(SR1_{SL})}{Tn_{ST}} * \mathrm{Com}_{ST} + \frac{\mathrm{Min}(SR2_{SL})}{Tn_{ST}} * Com_{ST} + \cdots + \frac{\mathrm{Min}(SRn_{SL})}{Tn_{ST}} * \mathrm{Com}_{ST}\right)}{n}
\end{aligned} \tag{6.16}$$

where:

- **n**: is the number of the selected security requirements for addressing a particular threat "Ti"

Then we calculate the average, as defined by Equation 6.17, of all the previously estimated effective values for each threat to determine a rough estimated value of the security achieved level for each component/asset.

$$\mathrm{Com}_{SA} = \frac{\sum_{k=0}^{m}(T1_{SA}) + (T2_{SA}) + \cdots + (Tn_{SA})}{m} \tag{6.17}$$

where:

- m: is number of all threats are impact a particular component/asset

We consider this estimation as a first indication of the achieved security level against the required security target level, which indicates how far the security target has been achieved. This result gives a preliminary value of the security achieved for a particular vehicular component/asset, which could assist the security architect to identify security gaps, which need more security concern.

6.6 Chapter Summary

This chapter includes a discussion on the metamodel of the proposed ontology-based cybersecurity framework. It gives a complete description of how the proposed framework applies multiple activities of each security phase. The data digestion phase is explained and discussed as the first activity applied by the framework to read and scan the input ontology representation model. The framework also applies a set of logical operations according to the theorem proving concept to verify the consistency among different ontology model entities. The validation concept is then discussed in this chapter to give a comprehensive overview of the validation process to validate the correctness of the selected security requirements against the existing security issues. The validation process followed in this work is based on the testbed execution approach to perform procedures for each vehicular component/asset individually to evaluate the correctness of security requirements. The chapter also discusses the importance of the proposed cybersecurity framework to address the identified security gaps and improve security by applying additional security requirements.

The next chapter represents the main case study used to demonstrate the novelty of the proposed cybersecurity framework for handling security issues in the ontology representation of a vehicular model. Outcomes of the research show the effectiveness of proposed ontological methodology for creating a complete path from threats to security requirements, which is not previously identified effectively. It also compares activities performed by the framework and the manual way. The chapter finally demonstrates that the proposed framework is complete automation actionable to save time and effort spent by a highly expert automotive security architect.

7 Use Cases and Experimental Model Evaluation

This chapter introduces an example of a real use case in the automotive domain to evaluate the reliability and validity of the proposed cybersecurity framework. The selected case study is a practical application for European automotive OEMs and automotive suppliers such as BMW, Audi, Volkswagen, Daimler, Bosch, etc. Additionally, for the autonomous vehicle developers include Waymo, Tesla, and others. This case study focuses on the automotive gateway, which is considered the first vehicle's defence line that needs more security protection capabilities to protect the rest of the vehicular elements from impacts of any external cyberattacks. In this case study, we aim to understand and clarify technical details related to this example to achieve high level cybersecurity goals to mitigate the identified potential threats and adapt risks to reach an acceptable level. In order to maintain legal obligations, names and some other details were changed to be generic.

This chapter includes content from some articles published within this thesis work. The following list refers to the selected publication that is integrated within the context of the chapter:

- *Shaaban, Abdelkader Magdy, Christoph Schmittner, Thomas Gruber, A. Baith Mohamed, Gerald Quirchmayr, And Erich Schikuta. "Automated Ontology-Based Security Requirements Identification for The Vehicular Domain." Journal of Data Intelligence 1, No. 4 (2020): 401-418.*

7.1 Case-Study: Potential Threats and Security Requirements Analysis of Components/Assets in the Automotive Domain

The system under consideration (SuC) combines a set of hardware and software for vehicles. The system consists of an Electronic Control Unit (ECU) as a processing unit induces different internal hardware components and a Linux-based operating system for vehicles. It provides different local and remote connectivity via multiple interfaces. The local connectivity is classified into the Ethernet as a wire interface unit and USB port. The Ethernet port is used for debugging, but in the future, it could be used for onboard connectivity. Some OEMs use the USB port for providing updates of their application, systems, or ECU units. The remote connectivity provides communication from the device to a wireless network through the WLAN (wireless interface).

Additionally, the system comprises a V2X unit to provide communication with external units (e.g., vehicles, RSU, and others). A Hardware Security Module (HSM) for V2X is integrated to provide a set of security protections. A communication element connects the central ECU unit with the other electric/electronic devices within the vehicle network throughout a communication element. The communication element manages the communication flow from/to the different connected devices. A communication ECU controls the communication between the other vehicular components described in the system. The system also provides a human interaction through a

Human-Machine Interface (HMI) that provides a touchscreen and multiple buttons. A conceptual design of this system is depicted in Figure 7.1, to describe the internal structure of the system.

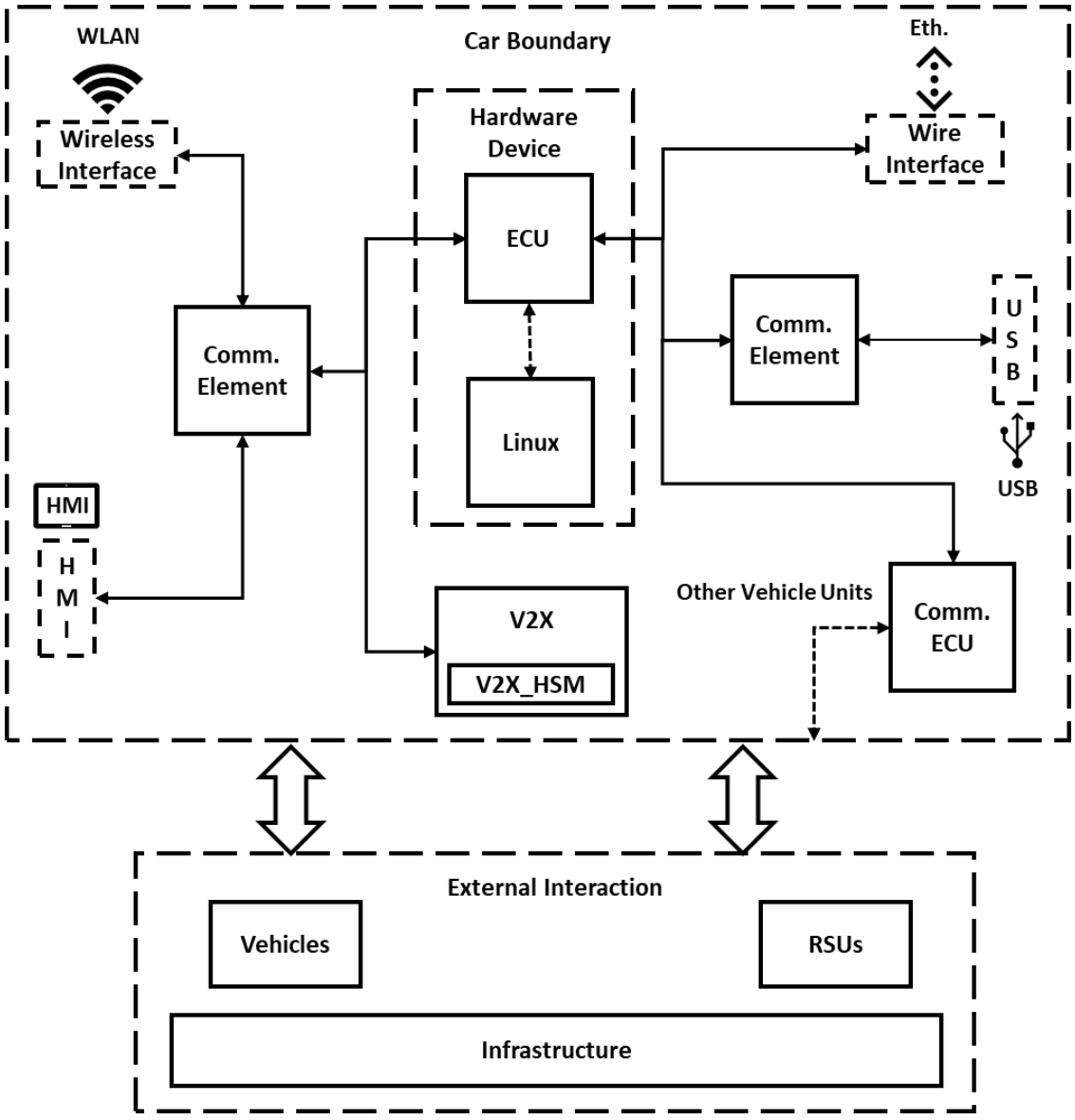


Figure 7.1: The conceptual design of an automotive case-study

As shown in the figure, a car boundary contains all internal components, and multiple communication channels connect these components within the vehicular network. Each unit contains a set of security properties expressed as a defensive mechanism to protect vehicle units from various malicious activities. The figure also describes communication between the vehicle's internal components with the external environment via the V2X unit, which may be another vehicle (connected vehicle scenario), RSU, or other terminals in the infrastructure. However, this example does not address the vehicle's surface interaction with external components. In this example, we create an ontology representation model that is identical to the model as described in Figure 7.1. The ontology model contains more details about relationships among all individuals in this example

and also all relevant aspects of each unit in the ontology structure.

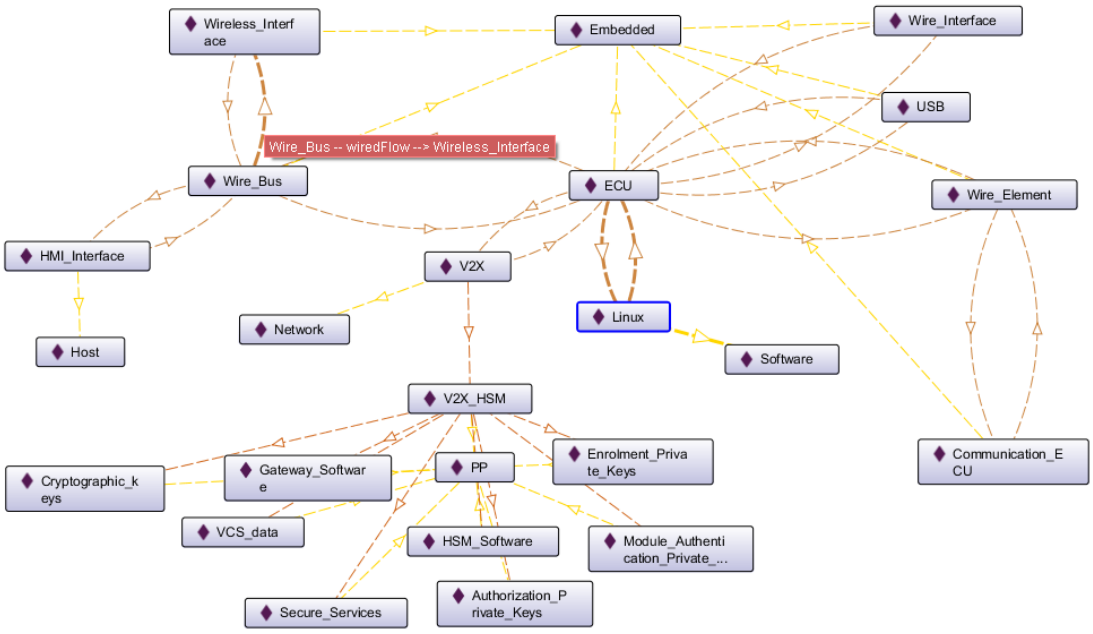


Figure 7.2: The use case illustration in an ontology representation

Figure 7.2 describes the ontology representation model of this case-study to illustrate the interactions among different individuals in this design. The proposed model accepts the ontology hierarchy and then applies a sequence of logical rules to the ontology model to determine whether or not security requirements can handle risks in a vehicle. A Java algorithm is built as a proof of concept of the proposed cybersecurity framework to study all data in the ontology model and discover security weaknesses in the design. Then it selects a set of security requirements that can fit to address existing security issues and also to fill the gap in case of the absence of the security requirements for the existing security issues.

As previously discussed, the proposed framework handles a wide range of potential threats based on the different resources, and also it has a set of security requirements from multiple resources. The framework uses the input data and checks whether or not of the selected security requirements can fit in addressing the existing security issues. This process is handled by applying a set of logical operations to verify and validate these requirements against security issues.

The proposed framework is built to accept various structures of the ontology inputs. There are four cases of inputs that can be handled by the proposed model to solve the existing security issues; these cases are defined as follows:

- Integration of both threats and security requirements.
- Import of potential threats.
- Import of security requirements.
- No remaining threats or uncovered security requirements.

Once the framework accepts the input, it checks all entities of the ontology form by applying a set of queries to understand the structures of the given ontology model and collect the existing data in the model to support in further processes. The framework uses the collected data to build a complete taxonomy of the vehicular example, to understand the interaction among a vast number of the ontological axioms that represent the input structure. The framework applies a set of rules to discover the existing security vulnerabilities and also identify new categories of potential threats that could have various impacts on the existing components. Then the framework selects applicable security requirements that match security properties of the existing threats with security measures of vehicular elements to handle these vulnerabilities and fill the gaps of non addressed weaknesses.

7.2 Experimental Framework

This case-study contains a wide range of security features that need to be taken into consideration while implementing and designing the ontology structure of this vehicular model. In order to save time and effort, we create the model in a spreadsheet and then create a set of rules that are able to translate and import the content of the spreadsheet into the required ontology structure. As mentioned and discussed in Chapter 4 and Chapter 5, the ontology structure shall be defined and structured clearly to avoid any mistakes on results regarding the poor structure of the ontology design. A set of Cellfire expressions are created to define the hierarchical structure of the ontology model and describe the consistency among various individuals within the ontology design. These expressions import the data identified in the spreadsheet and translate them into the ontology representation hierarchy. As discussed before, there are four scenarios of the data inputs that the proposed framework can accept. In this example, we combine these four categories of inputs within the ontology model to observe the effectiveness of the proposed framework and demonstrate that the proposed cybersecurity framework is fully-adaptable to handle multiple input forms.

Potential Threats: The ontology model could contain a set of previously selected potential threats are integrated within the ontology structure of the vehicular elements. The potential threats could be identified manually from any different resources such as the UNECE list of threats or could be identified by any threat analysis tool such as ThreatGet, as discussed in Chapter 2. In this case study, we used ThreatGet to identify and detect some potential threats relevant to a selected element in the input ontology model. Therefore, we select some selected outcomes from the ThreatGet to define and incorporate threats with the ontology model. This scenario aims to demonstrate the effectiveness of the proposed framework and examine its ability to manage threats that are not part of its threats KB or how the framework can handle threats that are obtained from external tools such as threat modelling approach. In addition, some other threats are selected manually from the Car-2-Car Protection Profile [Car18] [Car19]. We also used the UNECE threat list [UNE17b] to define some threats that could be incorporated into the input ontology structure to be processed by the proposed cybersecurity framework.

Security Requirements: In addition, the security requirements could be selected and integrated within the ontology model. The adaptability of the proposed framework can handle the existing components and select the applicable security requirements either they are previously selected or not identified. The framework defines individual sets of security requirements that are able to address particular security issue and protect a specific component from any malicious activities. The specified security requirements in the ontology model could be obtained either by the results

of any security requirement management tool such as MORETO as discussed in Chapter 2 or by selecting manually from any security requirement documents (e.g., common criteria [ISO09a], IEC 62443-4-2 security standards [IEC19]). In this example, we choose some security requirements from the IEC 62443-4-2 security standards for a selected vehicular unit. Also, regarding the content of the V2X-HSM protection profile, a list of security requirements are selected and integrated with the ontology input model.

Furthermore, to handle these different input scenarios, we designed a complete experiment that includes the four types of the ontology model. This model is defined as an input that needs to be processed and handled by the proposed cybersecurity framework. The framework applies a set of logical operations to the input ontology structure to verify the characteristics of the selected security requirements that are selected to protect the components/assets from different cyber malicious activities. Then the framework performs a test bed-execution based on the risk management process for each component/asset separately to validate these security requirements. The framework handles gaps discovered by the testbed validation process by identifying a new set of security requirements for managing existing security issues. In the following sections, the four separate input scenarios are addressed shortly to provide an overview of the selected component/asset combined with each input form to illustrate the effectiveness of the proposed cybersecurity framework.

7.2.1 Integration of both Threats and Security Requirements

The first input form aims to combine security requirements and possible threats with a selected component/asset into the ontology model. In this example, we use the "VCS data" asset and define the relevant security requirements and potential threats as described in the V2X HSM protection profile [Car18] and [Car19]. Figure 7.3 depicts this form of input structure to demonstrate the relationships between the "VCS data" asset, security requirements, and relevant potential threats.

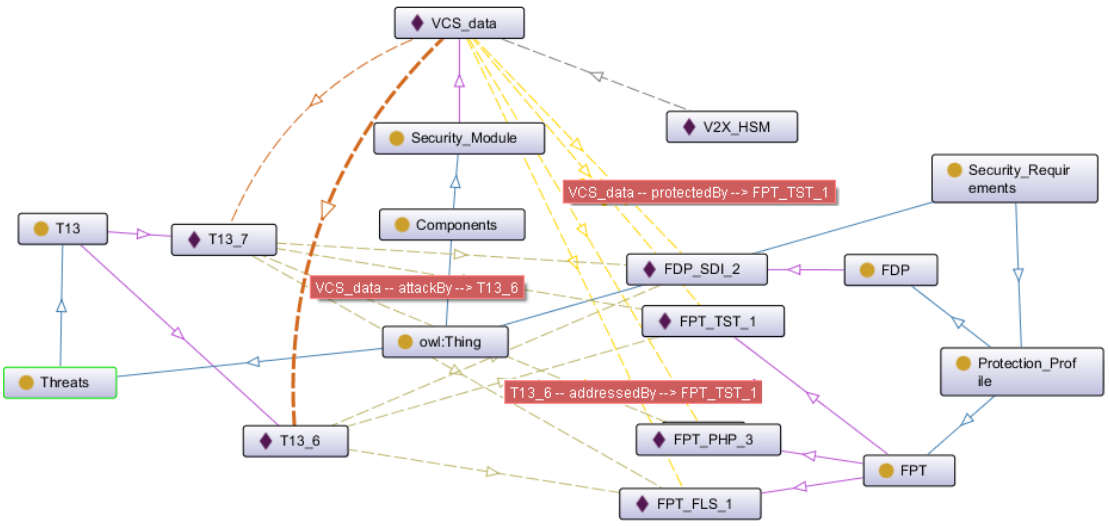


Figure 7.3: A set of selected security requirements and potential threats of the VCS data asset

The figure illustrates the interaction between the "VCS data" asset, security requirements, and relevant potential threats.

7.2.2 Import of Potential Threats

The ThreatGet tool is applied to the use case to identify the expected threats that have an impact on the defined components and assets. Selected results are integrated within the structure of the ontology model; the "attackBy" relationship is defined between threats and the impacted component/asset. We integrate some identified threats relevant to the ECU into the ontology model. In addition, some other selected threats from the UNECE threats list are selected manually and identified with this example to show that this type form contains a set of threats combined from multiple approaches (i.e., ThreatGet and manual way). Figure 7.4 illustrates that the ECU has a set of potential threats integrated within the ontology input.

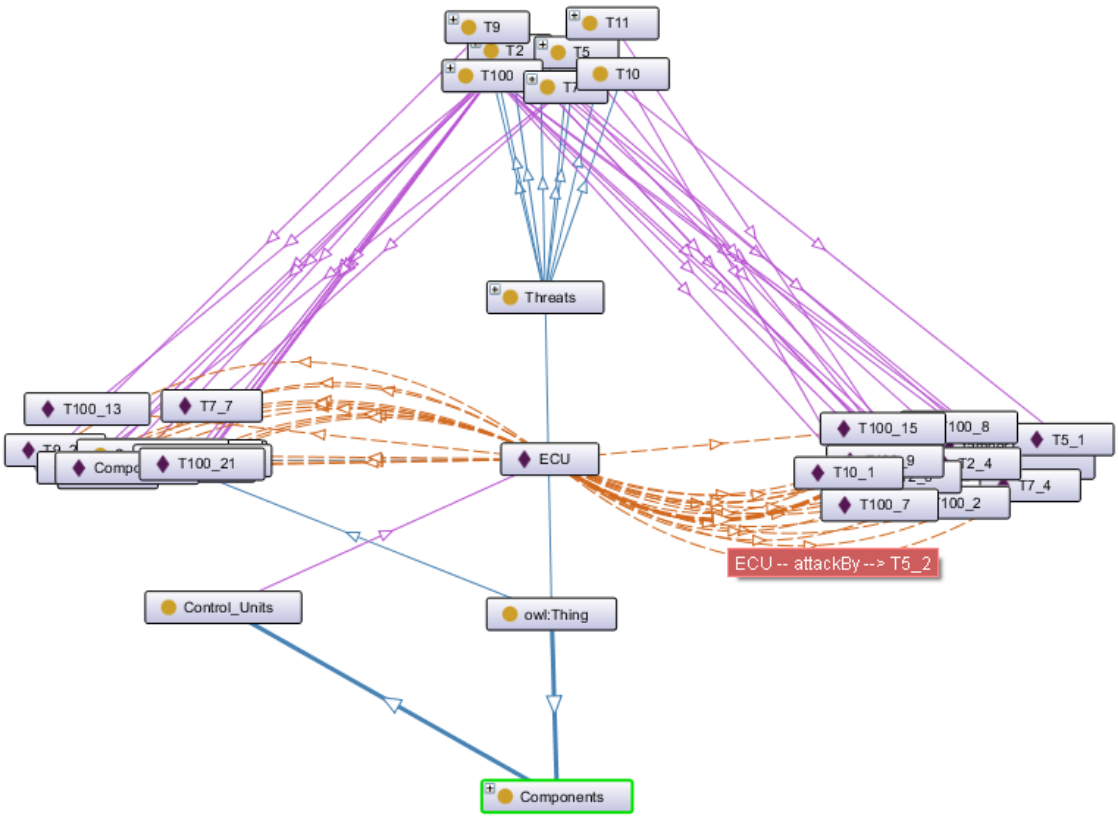


Figure 7.4: A set of potential threats for the ECU component

The figure shows the ECU component with the selected potential threats.

7.2.3 Import of Security Requirements

A collection of security requirements are chosen from IEC 62443-4-2 to be incorporated into the ontology structure as the third type of the ontology input that the proposed framework can handle. In this case, we select the V2X unit to check and demonstrate the effectiveness of the proposed cybersecurity framework against the applied security requirements as the only input is integrated within the ontology model. Figure 7.5 depicts the ontology structure when security requirements are stated as input within the ontology structure for a particular component such as the V2X unit.

As illustrated in the figure, the "protectedBy" relationship is defined between the V2X unit and

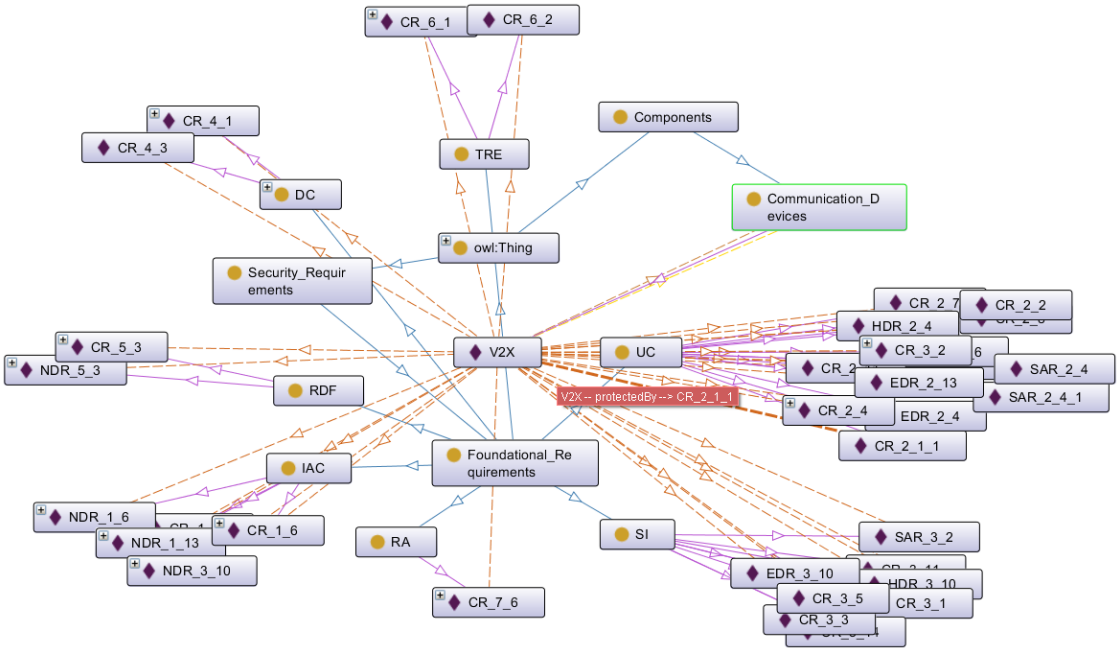


Figure 7.5: A set of selected security requirements for the V2X unit

the selected security requirements, which claim that these sets of requirements can protect the vehicle from different malicious attacks.

7.2.4 No Remaining Threats or Uncovered Security Requirements

The fourth type of input form defines a set of assets/components that define the ontology structure of the vehicular model without any knowledge about expected threats and relevant security requirements. In order to demonstrate the effectiveness of the proposed cybersecurity framework against this type of input, we use wire and wireless interfaces as selected vehicular components to be defined within the input. Then we combined all relevant security properties of these components. Figure 7.6 illustrates the ontological hierarchy of this example.

The figure shows the relationships between the selected vehicular components (i.e., wire and wireless interfaces) and their security properties. The "hasMitigation" relationship is defined to represent that each component has a set of security properties that represent protection mechanisms for reducing the risk.

7.3 Experimental Outcomes and Evaluation

We apply the cybersecurity framework to the previously discussed four examples to verify and validate the applied security requirements for each component/asset and address the discovered security gaps. Following sections discuss and evaluate outcomes of the cybersecurity framework for each input scenario.

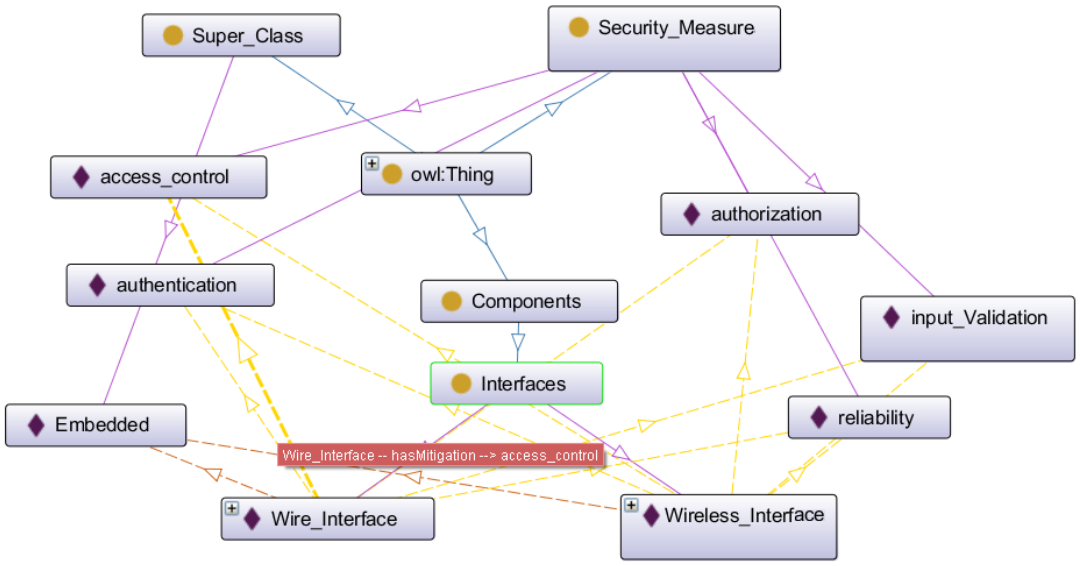


Figure 7.6: The ontology structure of the wire-wireless interfaces without any threats and security requirements

7.3.1 Case 1: The asset "VCS data" with the Integration of Potential Threats and Security Requirements

In this example, the asset "VCS data" has a set of potential threats and a set of security requirements are incorporated into this asset's ontology structure as an input to the cybersecurity framework. A collection of logical operations are applied to the ontology model to verify the accuracy of the asset's related ontology individuals and to check whether or not the selected security requirements are sufficient to protect it from various cyber-attacks.

These logical operations are applied to check the verification criteria of the selected security requirement(s) to address the given potential threats. The VCS_data asset has two threats T.VCS_DATA_MODIF and T.VCS_DATA_DISCLOSE, as depicted in Figure 7.3, these threats are defined for the selected asset (i.e., VCS data) according to the protection profile [Car19]. In this example, we re-named the abbreviation of the T.VCS_DATA_MODIF and T.VCS_DATA_DISCLOSE, to T13_6 and T13_7, respectively. Figure 7.7 illustrates an example of a logical rule that is applied to the ontology model to check if the threat 13_7, is addressed by security requirement(s) or not. Also, the rule checks if the same security requirements are defined to protect the asset of the identified potential threats.

As illustrated in the figure, the outcome is "true", which means all clauses in the verification rule are completely satisfied. The framework applies other sets of rules to verify the rest of the ontology entities relevant to the VCS asset such as T13_6 and relevant security requirements. After the verification procedures are completed, the framework launches the testbed execution process to validate the correctness of the selected security requirements and checks whether these requirements are fully fulfilled against the identified threats. As described in Chapter 6, the testbed process is built according to the risk management process, as discussed in ISO 27005 [iso11] based on ISO 31000. This method validates the accuracy of the selected security requirements for each component/asset individually.

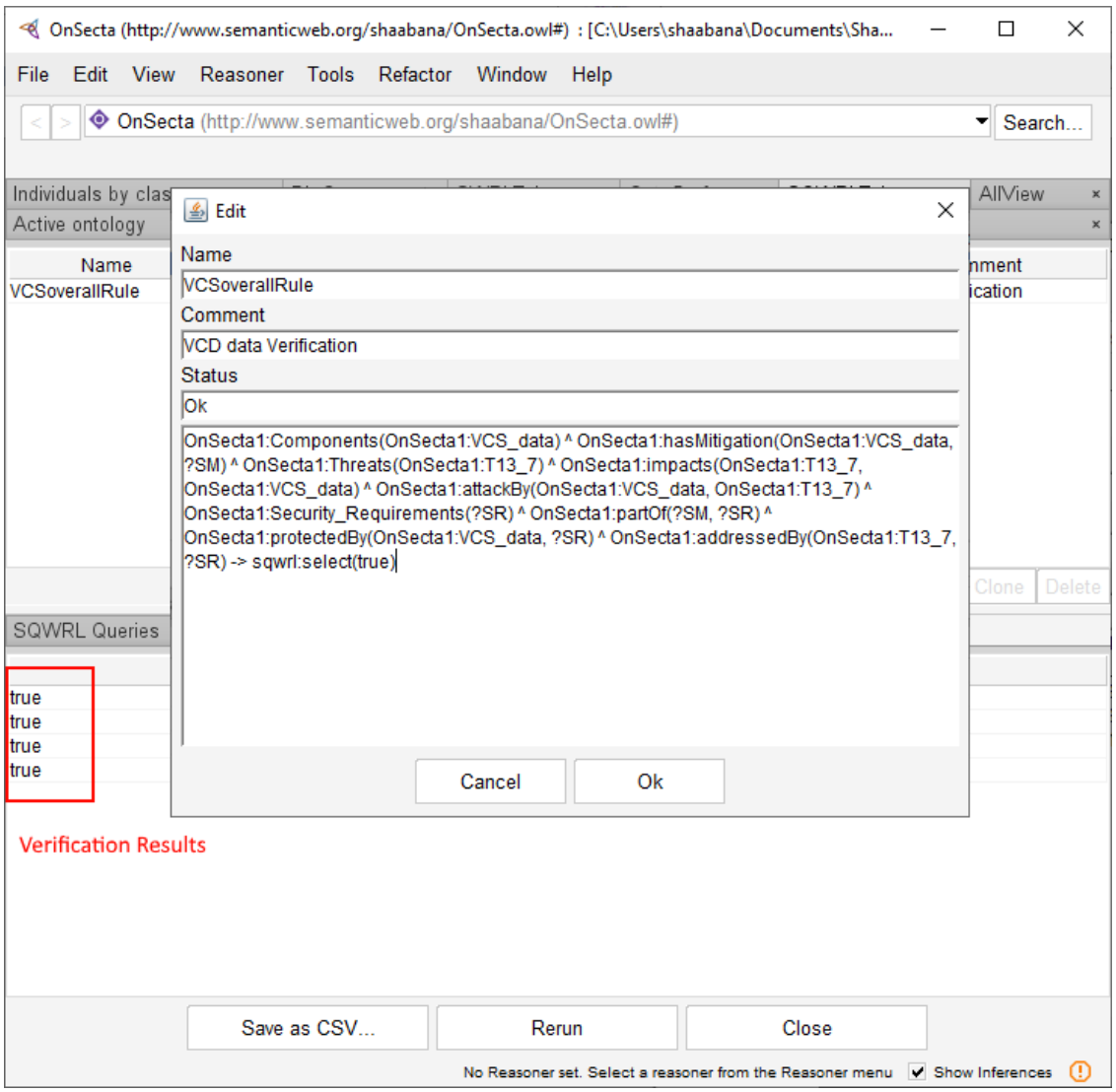


Figure 7.7: A verification rule for verifying the selected security requirements to protect the VCS data asset and addresses the exiting potential threats (e.g., T13_7)

The framework recognizes all possible potential threats that could affect the VCS data asset. This process is handled by the framework throughout the risk identification activity. Then it performs the risk analysis activity to detect threats that have a direct impact against the asset. The framework infers two threats (i.e., "T.VCS_DATA_MODIF" and "T.VCS_DATA_DISCLOSE"), which play a vital role in exploiting existing security weaknesses in the asset and allow attackers to use these vulnerabilities to perform various malicious activities. Once the framework infers threats that impact assets/components, it is essential to determine the risk of each detected threat; outcomes of this phase are used to establish the critical security goal level of the affected asset/-component. The risk degree of the previously identified threats (i.e., "T.VCS_DATA_MODIF" and "T.VCS_DATA_DISCLOSE") are evaluated as "Low" severity level, which means the security target level of the selected asset (i.e., VCS data) is equal to "1", as represented in Table 6.12. In order to achieve the requested target level of the asset, the framework applies a set of rules to

infer a group of security requirements that can address the identified threats and protect the asset (i.e., VCS data) from the negative consequences that could be generated according to these threats. The framework uses all security characteristics (i.e. asset security properties, threats information, component descriptions, security vulnerabilities data, risk degrees, and security levels) to infer the most closely matched security requirements to overcome these security weaknesses and meet the required security level.

Then the framework compares outcomes of this process with the previously selected security requirements to validate the selected security requirements against security weaknesses and protect the chosen asset/component. In certain instances, some of the previous security requirements might not be 100% matched to the inferred ones. Therefore, the framework examines the correctness of these security requirements against the inferred potential threats. It analyzes security characteristics of the selected security requirements and deduces a new collection of threats from the KB threats of the framework and then compares these threats to the threats previously inferred (i.e. in this case T13_6 and T13_7). This method aims to establish a reverse engineering operation from solutions to problems to understand the defence capabilities of the selected security specifications and to examine their effectiveness against the actual existing potential threats.

Figure 7.8 shows results of this process; the list of previously selected security requirements is specified in figure no. "1." Part no. "2" displays the list of derived security requirements by the proposed cybersecurity framework. These security requirements are selected according to the identified threat list, as shown in part no. "3", which describes the identified list of security threats identified by the framework in the risk identification process.

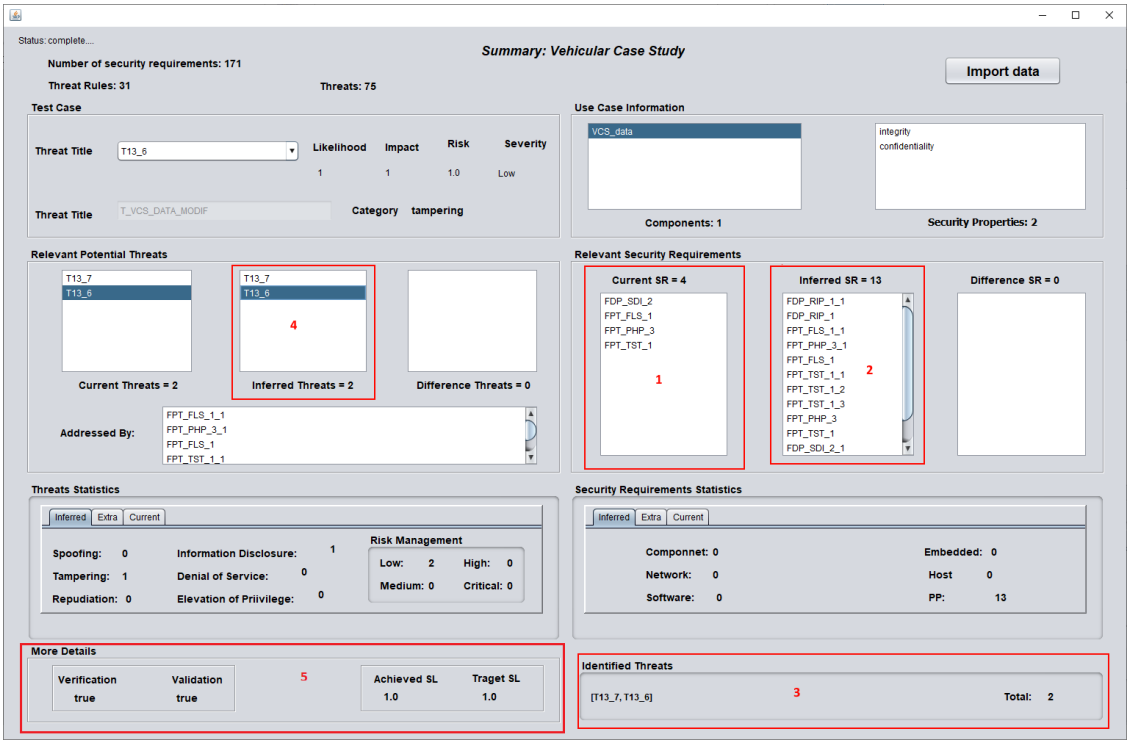


Figure 7.8: Outcomes of the cybersecurity framework according to the VCS_data asset

Then the framework deduces the exact potential threats that impact the VCS data asset, and displays them at part no. "4". The framework selects the relevant security requirements and

the other sub relevant ones that are defined in the PP according to the CC. The framework suggests these security requirements as a set of requirements that need to be integrated with the vehicular design to protect the vehicular asset (i.e., VCS data) from the identified potential threats (i.e., "T.VCS_DATA_MODIF" and "T.VCS_DATA_DISCLOSE"). Most of the inferred security requirements are defined as a subset of the previously selected security requirements, for this reason, the framework validates the newly inferred security requirements according to the previously selected security requirements, which are identified according to the protection profile [Car19]. Therefore, regarding results of the verification and validation processes, the selected security requirements are entirely applicable to address possible threats and protect the VCS data asset. The results of these processes are shown in Figure 7.8 part no."5". In addition, the achieved security level is also estimated based on the security level of the assumed security requirements. The framework estimates a preliminary estimation value that could help the system designer and demonstrate a simple numerical security level when these security requirements (i.e., the inferred security requirements) are implemented and incorporated within the asset structure in the vehicle network. Finally, the framework generates a set of test cases for each threat and related affected units (i.e. components/assets) to describe outcomes of all previous measures conducted by the cybersecurity framework. A simple example of the test case is described in Listing 7.1, as follows:

```
Threat: T13_6
Affected Unit: [VCS_data]
ID: A4_VCS_data
Components: VCS_data
Currents security requirements: [FDP_SDI_2, FPT_FLS_1, FPT_PHP_3,
    FPT_TST_1]
Inferred security requirements: [FDP_RIP_1_1, FDP_RIP_1, FPT_FLS_1_1,
    FPT_PHP_3_1, FPT_FLS_1, FPT_TST_1_1, FPT_TST_1_2, FPT_TST_1_3,
    FPT_PHP_3, FPT_TST_1, FDP_SDI_2_1, FDP_SDI_2_2, FDP_SDI_2]
Current Threats: [T13_7, T13_6]
Inferred Threats: [T13_7, T13_6]
Threats Identification: [T13_7, T13_6]
Difference Threats: []
Difference Security Requirements: []
Verification: true
Validation: true
Result: true
Achieved Level: 1.0
Target Level: 1.0
```

Listing 7.1: An example of the generated test case according to the inferred threats (T13_6) and relevant effected assets (VCS_data)

The test case is generated according to each threat and all details of the affected vehicular unit (i.e., selected threats, inferred threats, selected security requirements, inferred security requirements, and the results of the verification and validation). According to the protection profile [Car19] and the inferred results of potential threats, there are no other affected units of the T13_6 except the VCS_data asset. Our findings on the VCS_data asset, according to the protection profile [Car19], this asset is impacted by two threats. The first threat is T.VCS_DATA_MODIF

Table 7.1: Comparison between the inferred security requirements by the proposed cybersecurity framework and the identified ones based on the V2X HSM protection profile

Potential Threats	Inferred Results	PP Security Requirements
T.VCS_DATA_MODIF	FDP_SDI_2 FDP_SDI_2_1 FDP_SDI_2_2	**FDP_SDI_2**
	FPT_TST_1 FPT_TST_1_1 FPT_TST_1_2 FPT_TST_1_3	**FPT_TST_1**
	FPT_PHP_3 FPT_PHP_3_1	**FPT_PHP_3**
T.VCS_DATA_DISCLOSE	FPT_FLS_1 FPT_FLS_1_1	**FPT_FLS_1**
	FDP_RIP_1 FDP_RIP_1_1	**None**

(is named T13_6 in this experiment), which exploits the asset's integrity. The second threat is T.VCS_DATA_DISCLOSE (that is renamed to T13_7 in this work), which breaches the confidentiality of the asset.

Case 1: Evaluation

In the protection profile [Car19] there are a set of security requirements that are intended to address the identified threats and to protect "VCS data" asset. These security requirements are defined and selected according to the common criteria. The framework manages these security requirements to check the input ontology model and to identify the proper set of security requirements to fill security gaps in the system model. Our framework analyses the input ontology model and performs a set of logical rules to correlate the input with relevant security requirements. A set of security requirements have been inferred according to the common security measures between the security requirements, assets, and potential threats. This set is considered a group of suggested or recommended security requirements that assist the system designer in selecting and finding an appropriate set of security requirements to fulfil security gaps in the vehicular design. The framework finds 13 security requirements are defined and explained in the protection profile [Car19]. Table 7.1 illustrates the obtained results by the proposed framework. Most of the results are compatible with security requirements that are defined in the protection profile [Car19], where only one security requirement and the relevant sub-requirement (i.e., FDP_RIP_1 and FDP_RIP_1_1) are deduced from the security requirements KB. We cannot claim this value is wrong because when re-evaluating them, we found some matching characteristics between these security requirements and the T13_7 threat.

Therefore, we can conclude that the framework focuses on the most appropriate security requirements that contain common security measures with threats and particular assets/components. This process is achieved successfully by the proposed framework in a short time. The second column (i.e., Inferred Results) in Table 7.1 represents the set of identified security requirements

by the proposed framework, where the third column (i.e., PP Security Requirements) represents the security requirements that are defined in the protection profile [Car19]. Most of the results lead to a similar matching between the framework's results, and security requirements that are defined in the protection profile. The other identified security requirements (i.e., FDP_RIP_1 and FDP_RIP_1_1) are defined because they have common properties between the asset (i.e., VCS data) and the potential threats (i.e., T.VCS_DATA_MODIF and T.VCS_DATA_DISCLOSE). The system or vehicular security architect could check these results of the identified security requirements to decide which could be selected or ignored.

7.3.2 Case 2: The ECU Component with Importing of Potential Threats

The second scenario of the ontology input could be represented as a set of potential threats for particular component(s) or asset(s). In this example, we apply the ThreatGet tool to the vehicle example (i.e., as described in Figure 7.1), and collect outcomes of the tool for a vehicular unit to be part of the ontology input model. As illustrated in Section 7.2.2, we select the ECU component in this example to demonstrate the efficiency of the proposed framework for handling and addressing the external threats collected by any threat analysis approaches by proper security requirements. In this case, the framework will perform the testbed process to infer specific security requirements based on the inferred potential threats. The framework also manages the other potential threats identified by external tools, such as the ThreatGet tool. These external threats are categorized as "different potential threats." The framework uses the same approach to recognize the existing security vulnerabilities and assess which security weaknesses could be exploited. This helps in giving the most applicable security requirements that can address existing threats correctly.

The new inferred list of potential threats are defined according to their impacts against the ECU component; this list is depicted in Figure 7.9. As discussed before, this example does not contain any security requirements; hence the framework aims to apply a new set of rules to deduce sets of security requirements that can address the newly inferred threats, and the other ones are defined within the input ontology model. Each threat that should be addressed by at least one appropriate security requirement, so as depicted in the figure, the "Addressed By" field represents all inferred security requirements for a particular selected threat. For example, threat "T5_2" is identified by the cybersecurity framework; this threat exploits some security vulnerabilities in the ECU unit to allow attackers to perform malicious activities. Therefore, the framework detects two security requirements (e.g., EDR_2_4, and EDR_2_4_1) to address this threat. All detected security requirements are depicted in Figure 7.10.

As shown in Figure 7.10, the framework selects the most applicable security requirements of the identified security issues while considering the category of the impacted unit. As discussed in the earlier chapters, the IEC 62443-4-2 classified the components into four categories (e.g., software application, embedded device, host device, and network device); however, the majority of the defined security requirements in [IEC19], are the same for these types of components (i.e., component requirement). The framework performs the inference rule execution based on the category of the security requirements and its similarities with the impacted unit classification. In our case, the ECU unit is classified as an embedded device and framework succeeded to select the applicable security requirements for this category for solving the current security issues. However, one component requirement (CR) is selected to solve a security issue that cannot be solved by the EDR categories. The framework starts with focusing on the applicable security requirements regarding the category of the impacted unit. Hence it starts on a small-scale by applying the inference rules. Then if no applicable security requirements are defined, it starts to increase the

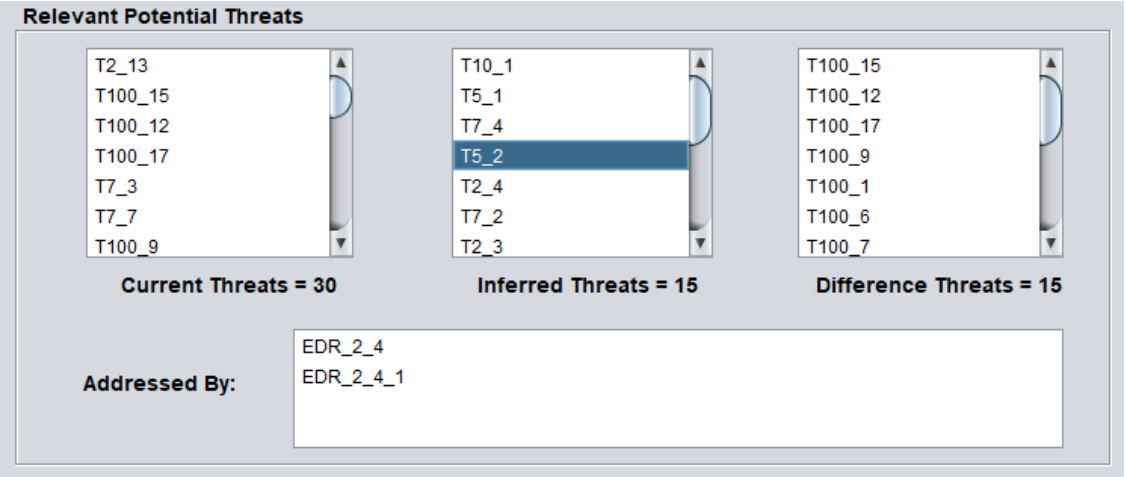


Figure 7.9: Lists of all potential threats of the ECU

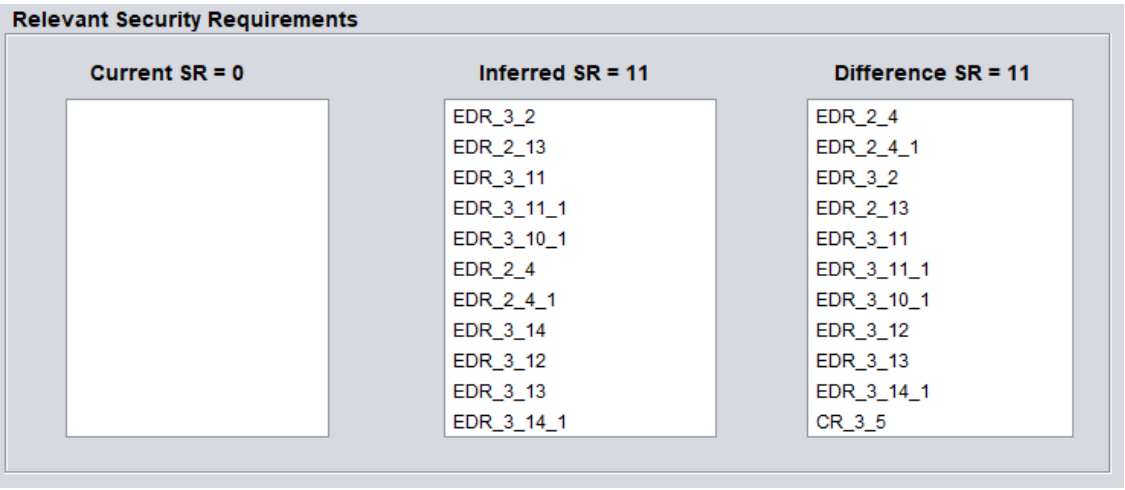


Figure 7.10: Relevant security requirements for addressing the security issues of the ECU unit

inference scope into the main category (i.e., component requirements) to find the applicable ones. In order to demonstrate that, we perform the same experiment on the existing potential threats of the ECU unit with the lack of embedded security requirements categories (i.e., EDR) in our security requirements KB. Figure 7.11 illustrates the results of the same list of threats that are depicted in figure 7.9. However, the only difference is that the framework deduces a new set of security requirements to address the threat T5_2.

Figure 7.11 illustrates outcomes of the inference process, which elect a distinct collection of security requirements according to the most relevant existing security requirements. Figure 7.12 shows the new selected list of all inferred security requirements for the ECU component.

Case 2: Evaluation

The two conducted experiments show that the obtained outcomes refer to the framework's ability to find the most applicable security requirements to handle existing security issues. Table 7.2 illustrates a comparison between the two experiments that applied to the ECU component.

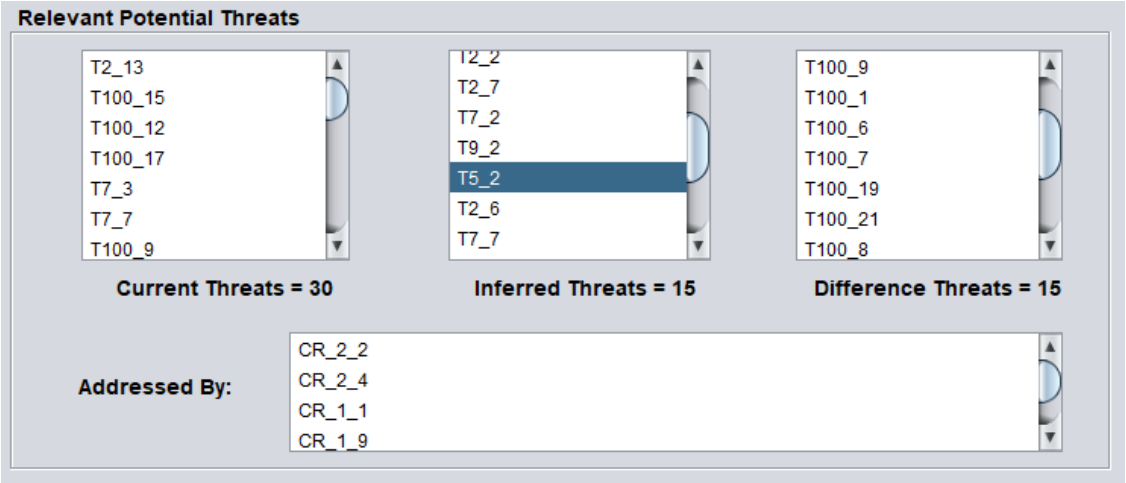


Figure 7.11: Lists of all potential threats of the ECU in the absence of the EDR security requirements

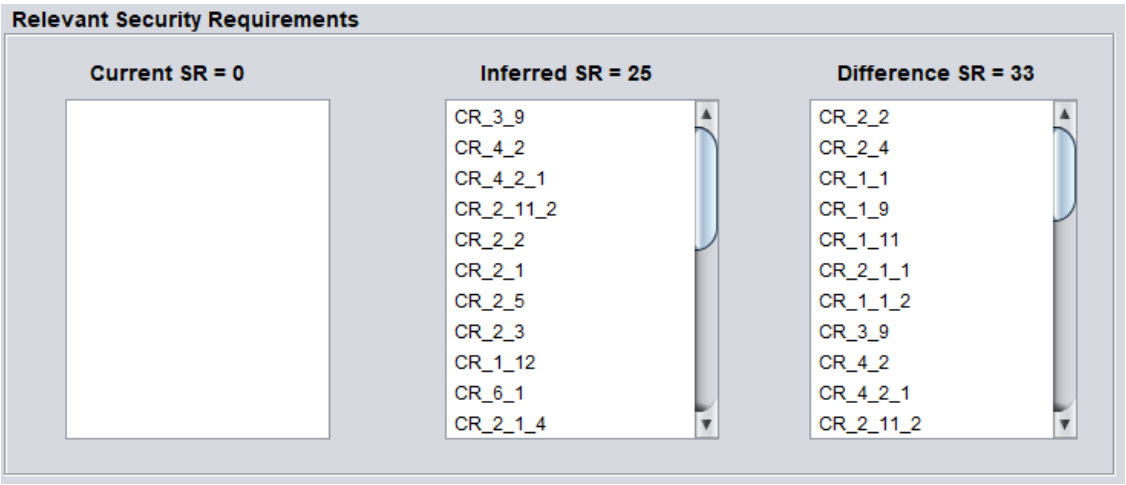


Figure 7.12: Relevant security requirements for addressing the security issues of the ECU unit based on the CR categories

The table represents the calculation process of the achieved security level for the ECU component. The process starts by performing the risk evaluation process for each threat according to the likelihood and impact values. The risk outcome values are used for each threat to represent its severity level. The maximum severity level of all threats is defined as the security target level of the ECU component. The severity level is evaluated according to Table 6.12. Then the ECU components' target level is defined as "4" as defined in Table 7.2. The values in the column "1st Exp." represent the achieved security level according to the selected security requirements based on the EDR category. The column "2nd Exp." presents the achieved security level after some other relevant security requirements from the general categories (i.e., CR) are selected to address security issues. Figure 7.13 illustrates the rate of the suggested security requirements for each threat to address the existing security weaknesses. These two categories of the selected security requirements are illustrated in the figure, to see how the framework selects the most

Table 7.2: Comparison of security levels achieved between the first (existence of EDR) and second (absence of EDR) experiments are applied to the ECU

#	Inferred Threats	Likelihood	Impact	Likelihood x Impact	1st Exp.	2nd Exp
1	T10_1	1	4	4	2	2
2	T5_1	4	2	8	3	0
3	T7_4	1	3	3	1	0
4	T5_2	4	2	8	3	3
5	T2_4	4	3	12	4	4
6	T7_2	1	3	3	1	0
7	T2_3	4	3	12	4	0
8	T2_13	2	3	6	2	2
9	T7_3	2	3	6	2	2
10	T7_7	1	4	4	2	0
11	T2_2	4	3	12	4	0
12	T9_2	1	1	1	1	1
13	T2_7	4	3	12	4	0
14	T2_6	4	4	16	4	0
15	T2_12	4	3	12	4	4
1st Experiment - Achieved Level					**4**	
2nd Experiment - Achieved Level					**1.866**	

closely matched security requirements that could be used to identify the protection mechanism against the identified threats.

The figure demonstrates that selecting other security requirements could help address the current security issues in the absence of the highly recommended ones. As depicted in the figure, some threats are not addressed regarding the undefined suitable security requirements for exiting security issues. The framework evaluates the ECU's security level after applying the security requirements from both EDR and CR groups. The evaluation of the achieved security level is different in these two situations. Each completely addressed threat contains a set of suggested security requirements that can increase the current security level of a threat to the target level. However, the non-addressed ones will still be an issue in the system model. The framework gives an estimation value for this issue to clarify that these selected security requirements are not sufficient to reach the foremost security goal. Table 7.2 gives estimation values of the achieved levels for both experiments. The 1st experiment is achieved to "4", because all threats are well-addressed by the applicable security requirements. The 2nd experiment is achieved to "1.86", regarding the non-addressed threats in this case.

As illustrated in Figure 7.14, the orange points represent the achieved level of each threat according to the security target value (in square bracket), according to the EDR category. The blue line represents the achieved security level for selecting the CR groups and the lack of EDR category.

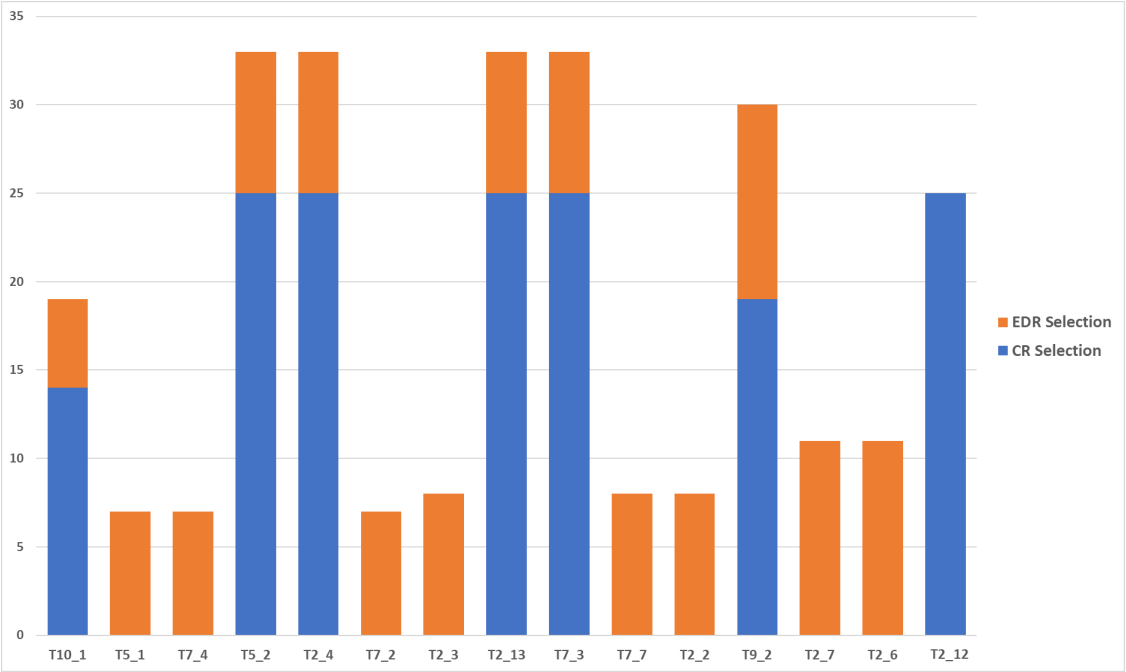


Figure 7.13: The number of the selected security requirements according to the CR and EDR security categories

7.3.3 Case 3: The V2X Unit with Importing of Relevant Security Requirements

For this case, we use the V2X unit and define some selected security requirements that could be applied to define a suitable set of security protection mechanisms against different attack scenarios. The chosen security requirements are selected without any prior knowledge about the exact potential threats. That means these requirements could be redundant or not applicable for addressing security issues. To select security requirements randomly without any particular risk identification or analysis plan is considered a real security issue, which could lead to unwanted consequences. Therefore, to create a secure system, we need to know the exact security weaknesses and then determine which precise security requirements are needed to address these system design issues. From the cost point of view, selecting precise solutions for the particular security issue could be more cost-effective than applying random ones.

Therefore, the framework applies a set of procedures to V2X individuals in the ontology model to check the correctness of the selected security requirements. Regarding the outcomes of the risk analysis and risk evaluation processes, the framework deduces and suggests a set of security requirements. These requirements are defined as a suggested group of security requirements that could be decided by the system design to protect the vehicular assets/components from identified security issues. Figure 7.15 illustrates a list of the inferred potential threats of the V2X unit; where Figure 7.16 depicts the relevant security requirements that are selected according to identified threats.

The inferred security requirements are considered guidance to the system design that follows to determine which of these could be part of the vehicle design process. The previously selected security requirements defined within the input will be a part of the framework's outcomes. Any

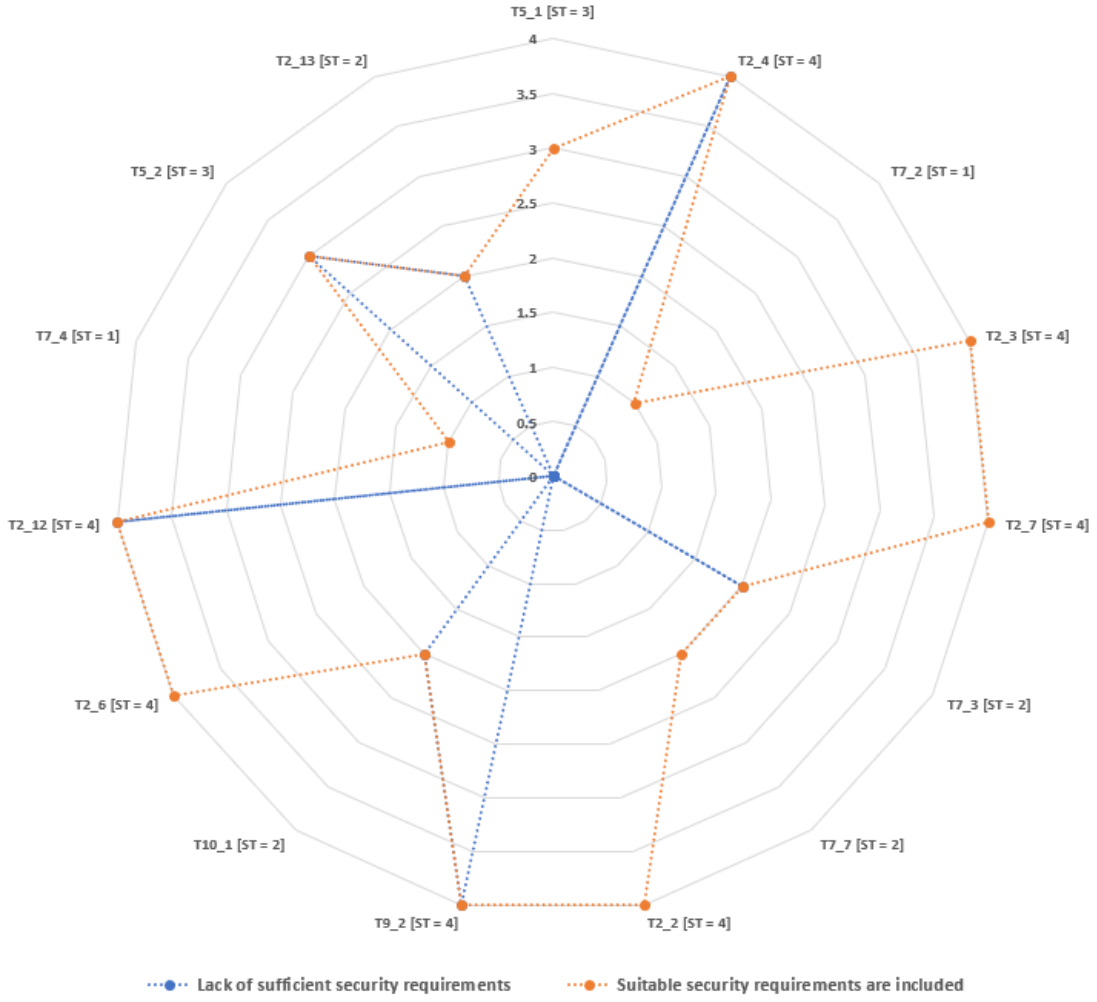


Figure 7.14: Achieved security level of each threat according to the selection of EDR (orange) and CR (blue) security requirements categories

other security requirements management methodology selects a set of security requirements. Part or all of these requirements could be sufficient to address existing security issues. In order to check that, the framework uses all these requirements to validate the selected requirements in addressing the existing security issues. It uses security characteristics of the selected security requirements that define a set of potential threats to be compared with the inferred ones.

Case 3: Evaluation

Security requirements within the ontology input are selected without any knowledge about the current security issue in the given model, so we could expect that some redundant or ineffective requirements are integrated. The inferred list of requirements is defined according to outcomes of the risk analysis process applied by the proposed framework. Therefore, the framework only selects the security requirements according to the identified threats. The framework performs a reverse action to deduce a new set of potential threats from the input's integrated security

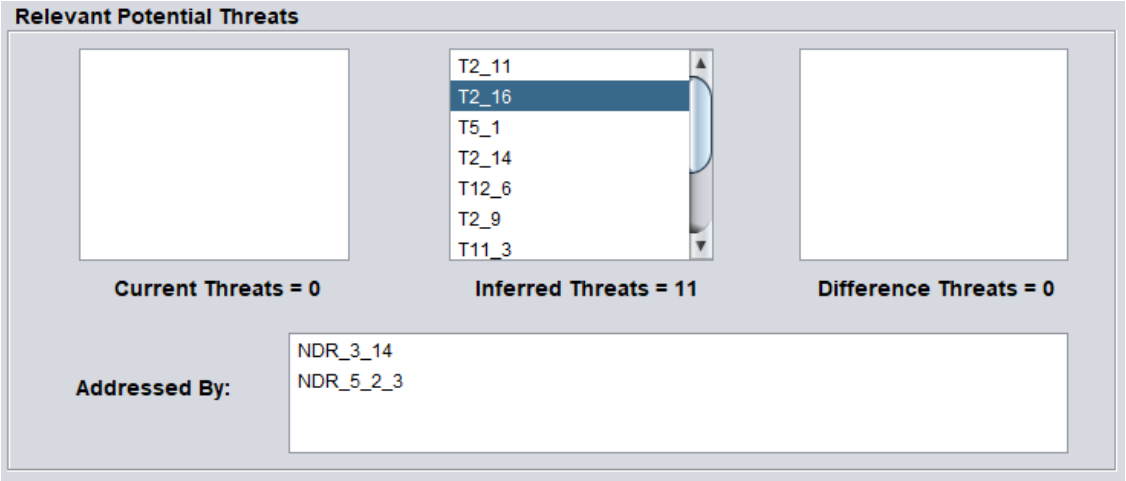


Figure 7.15: Lists of the inferred potential threats of the V2X unit

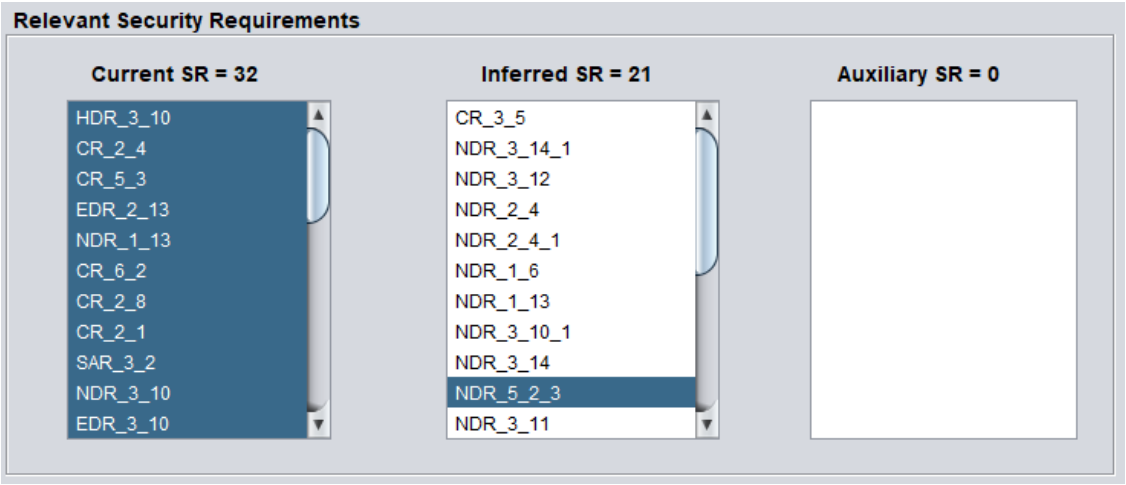


Figure 7.16: Relevant security requirements for addressing security issues of the V2X unit based on the CR category

requirements to test their effectiveness against potential threats. Table 7.3 represents outcomes of the validation for each selected security requirement against the previously inferred potential threats.

As defined in the table, some of the selected security requirements are insufficient to address the existing weaknesses. For example, some of these security requirements have an ineffective impact against the inferred potential threats; some others have low effectiveness toward identified threats, such as some security requirements affecting roughly on 9% of the overall threats and others are over 50% of threats. The values in the table evaluate the effectiveness of each security requirements according to the matching outcomes of newly inferred potential threats based on the extracted security properties from each security requirements against the previously selected potential threats according to the outcome of the risk analysis process as depicted in Figure 7.15. The overall evaluation is not validated because no selected security requirements are applicable to address the threat (T11_4).

Table 7.3: Percentage of effectiveness for the selected security requirements against the inferred threats

#	Security Requirements	Inferred Potential Threats											Rate of Effectiveness	Validity	Percentage of Effectiveness
		T11_3	T2_11	T11_1	T5_1	T6_1	T2_16	T6_3	T2_14	T11_4	T12_6	T2_9			
1	HDR_3_10												0	FALSE	0%
2	CR_2_4				X	X							2	TRUE	18.18%
3	CR_5_3												0	FALSE	0.00%
4	NDR_1_13	X		X		X	X	X	X				6	TRUE	54.55%
5	EDR_2_13	X											1	TRUE	9.09%
6	CR_6_2												0	FALSE	0.00%
7	CR_2_8												0	FALSE	0.00%
8	CR_2_1	X		X		X	X	X	X				6	TRUE	54.55%
9	SAR_3_2				X								1	TRUE	9.09%
10	NDR_3_10												0	FALSE	0.00%
11	EDR_3_10												0	FALSE	0.00%
12	CR_3_5		X						X		X		3	TRUE	27.27%
13	NDR_1_6			X		X	X	X	X				5	TRUE	45.45%
14	CR_3_1			X			X				X		3	TRUE	27.27%
15	EDR_2_4				X	X							2	TRUE	18.18%
16	CR_3_14												0	FALSE	0.00%
17	CR_1_6												0	FALSE	0.00%
18	CR_1_11			X	X	X	X	X	X				6	TRUE	54.55%
19	CR_2_6												0	FALSE	0.00%
20	CR_3_11				X								1	TRUE	9.09%
21	CR_2_2	X			X	X							3	TRUE	27.27%
22	CR_4_1											X	1	TRUE	9.09%
23	CR_2_11												0	FALSE	0.00%
24	SAR_2_4				X	X							2	TRUE	18.18%
25	CR_7_6				X								1	TRUE	9.09%
26	HDR_2_4				X	X							2	TRUE	18.18%
27	CR_3_3												0	FALSE	0.00%
28	CR_4_3												0	FALSE	0.00%
29	NDR_5_3	X											1	TRUE	9.09%
30	CR_6_1	X											1	TRUE	9.09%
31	CR_2_7												0	FALSE	0.00%
32	CR_2_1_1			X	X	X	X	X	X				6	TRUE	54.55%
Addressed		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes			
Overall Evaluation		FALSE													

The framework demonstrates that it is scalable to accommodate different types of inputs. It can also infer the most applicable security requirements that could be matched entirely with the existing security issues. The framework shows that accepting various input forms in a scalable manner. It can also infer the most applicable security requirements that could completely match current security problems. The addressed problem in this input form is that the selected security requirements are defined without knowing the exact security issues. That could lead to inadequate or redundant security requirements; where the proposed system starts by identifying the expected security threats, then defines relevant security requirements. To demonstrate that, we apply security properties on the V2X unit according to the security requirement and repeat the analysis process once again. The results of these experiments are illustrated in Figure 7.17.

The blue bars represent the previously selected security requirements. The orange bars illustrated the number of selected security requirements when no change happened on the security properties of the V2X unit. We apply the security authentication mechanism on the V2X unit and then apply the experiment once again. The grey bars show the rate of the selected security requirements according to the change of the applied security properties. Then we apply two more security properties into the V2X unit; we get more concentrated security requirements for the still existing security issues. For each new change, we receive a set of security requirements according to the real existing security issues, which means the framework only selects the security in case it is required to protect a partial vehicular unit. The figure presents results according to the foundational security requirements classification as discussed and presented in [IEC19]. The classifications of the foundational requirements are discussed earlier in Section 4.1.1.

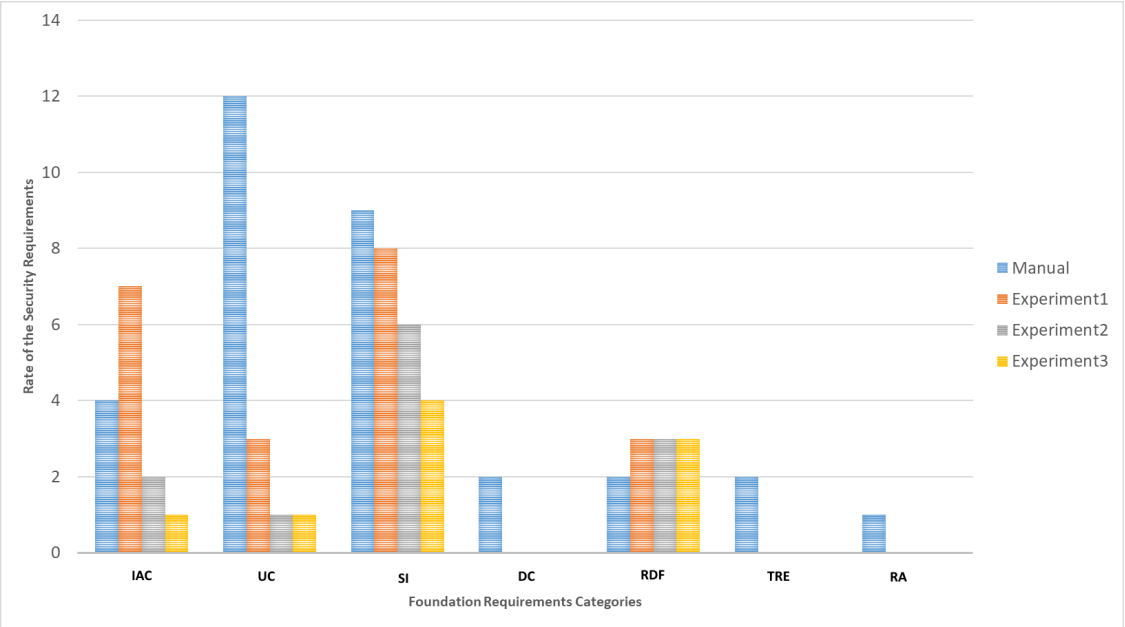


Figure 7.17: Comparison between the security requirements that previously selected and the impact of selecting security requirements based on changing security properties for the V2X unit.

7.3.4 Case 4: The Wire/Wireless Interfaces Without Remaining Potential Threats or Uncovered Security Requirements

In this example, we use the wired and wireless interfaces defined in the case study structure in Figure 7.1. In this input scenario, we only define interfaces with relevant security properties, and then apply the cybersecurity framework to the ontology model. Regarding the absence of both the security threats and requirements, the framework starts defining the common potential threat that could impact these interfaces. Then, it performs the risk analysis process to check whether or not any of the previously identified threats in the risk identification process impact against the selected vehicular units.

Case 4: Evaluation

Unlike the other previously discussed cases, there are no threats and security requirements defined in this case. Therefore, the framework, in this case, identifies all potential threats and suggests all relevant security requirements can address the identified security issues. Table 7.4 illustrates the outcomes of the framework for identifying potential threats and suggesting relevant security requirements.

The table shows the outcomes of the proposed model according to the given input. Results contain a set of potential threats that impact interfaces and a set of appropriate security requirements for addressing these threats. The framework can analyze the given input and identify security issues. Then it applies a set of logical rules to deduce the precise security requirements.

Table 7.4: The outcomes of the proposed framework according to the given input

Vehicular Components	Wire_Interface	Wireless_Interface
Potential Threats	T6_5 T1_1 T2_1	T6_5 T6_1 T1_1 T6_3
Security Requirements	EDR_3_11 EDR_2_13 EDR_3_2 EDR_3_11_1 EDR_3_10_1 EDR_2_4 EDR_2_4_1 CR_2_2	EDR_3_11 EDR_2_13 EDR_3_2 EDR_3_11_1 EDR_3_10_1 EDR_2_4_1 EDR_3_12 EDR_3_13 EDR_3_14_1

7.4 Evaluation Summary

Ontologies are considered a powerful method that uses regular specifications for knowledge representation such as vocabularies, taxonomies, classes, individuals, and annotations. Ontologies function like the human brain. They work and reason with concepts and relationships among multiple entities. That is considered the same way as humans perceive interlinked thoughts [Ont18]. Ontologies are integrated with this proposed model to perform security verification and validation in the vehicular domain. The vehicle development process requires the merging of a significant number of security requirements from multiple resources. For instance, the requirements that relate to the Security Development Lifecycle (SDL) are appropriate to all industrial applications such as vehicle development [McA16]. Managing hundreds or thousands of security requirements is considered a challenging task because it is time-consuming, and complex work needs a sufficient cybersecurity experience level. The structure of the ontologies has a significant role in reducing complexity [CC10]. Furthermore, the framework manages ontologies by applying rules over a massive number of ontology entities and defining relationships and concepts matching new security requirements to achieve a particular security level.

The framework is designed to be fully-adaptable to handle different forms of ontology inputs representing vehicular components/assets interactions with security protections and their relations with the applied security requirements. The input ontology model could contain a set of previously selected potential threats that have a negative impact on the existing vehicle units. Therefore, in our case study, we create four different experiments for evaluating the effectiveness of the proposed cybersecurity framework for checking and handling the existing security issues in the vehicular design. These types of input forms are defined as follows:

- **Case 1:** Integration of both threats and security requirements.
- **Case 2:** Import of potential threats.
- **Case 3:** Import of security requirements.
- **Case 4:** No remaining threats or uncovered security requirements.

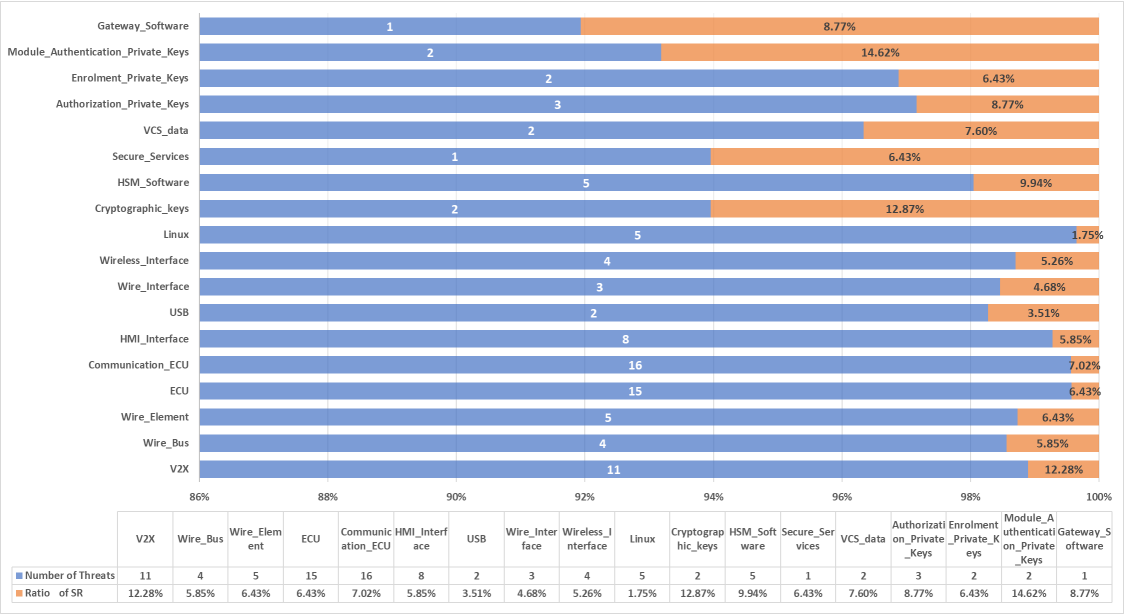
Case 1: The first experiment demonstrates the framework's ability to verify and validate the compliance of the selected security requirements on the vehicular asset (i.e., "VCS data"). It infers new security requirements from the security knowledge-base (KB). These requirements are extracted based on a set of calculations performed by the proposed framework on the input ontology model. Outcomes prove the conformity between the newly selected security requirements and the previously selected security requirements described in the V2X HSM protection profile [Car19]. This experiment also confirms that the framework can handle the vehicular assets, which defines a critical unit with a high value from the attacker point-of-view.

Case 2: The framework demonstrates its efficiency in importing threats from external threat analysis approaches, as explained in the ECU unit in Section 7.3.2. In this case, the framework selects all relevant security requirements similar to the same category of the vehicular component (i.e., embedded in this case). When we change the strategy of this experiment and keep only the component security requirements (i.e., CR), we notice that the framework also selects the applicable set of security requirements even when lacking of the completely matched one (i.e., EDR). That means the framework can select the security requirements according to existing security requirements in the knowledge-base. Therefore, the knowledge base plays an essential role in this process; accordingly, the KB shall be kept up-to-date with the most applicable security requirements relevant to the automotive domain.

Case 3: In the case of V2X unit, the framework plays a significant role in validating the correctness of the applied security requirements, which are defined by an external tool or by a manual approach. The framework deduces new security issues from the existing security solutions (i.e., security requirements), which performs as reverse engineering action for testing the effectiveness of the previously selected security requirements against the particular existing security issues.

Case 4: In the case of both potential threats and security requirements not being identified within the input. The framework also acts efficiently for filling the existing security gaps in the vehicular design. It defines security issues for each component and selects the appropriate set of security requirements to protect vehicular units and address the identified potential threats.

Although the inference methodology takes time for deducing new results, the overall evaluation of the performance is relatively faster than any comparivate manual approaches. The framework can save much time and effort spent by an expert person to verify and validate the applied security requirements. It is also quite a challenging process for selecting a set of applicable security requirements for protecting the vehicular components/assets against a particular security issue. Our case study has about 18 vehicular components/assets; each has a set of potential threats and needs a group of security requirements to protect it. According to our security requirements knowledge-base, we have 171 security requirements collected from different resources, as discussed previously in detail within this work context. Also, we have 75 threats and 124 rules representing the behaviour of these threats against the vehicular components/assets. In order to select the most applicable security requirements from the KB to fit in this example, we could not guarantee the correctness and completeness of the outcomes. Figure 7.18 shows the number of inferred threats and the ratio of the selected security requirements for each vehicle component/asset out of the existing security requirements in the knowledge base.



	V2X	Wire_Bus	Wire_Element	ECU	Communication_ECU	HMI_Interface	USB	Wire_Interface	Wireless_Interface	Linux	Cryptographic_keys	HSM_Software	Secure_Services	VCS_data	Authorization_Private_Keys	Enrolment_Private_Keys	Module_Authentication_Private_Keys	Gateway_Software
Number of Threats	11	4	5	15	16	8	2	3	4	5	2	5	1	2	3	2	2	1
Ratio of SR	12.28%	5.85%	6.43%	6.43%	7.02%	5.85%	3.51%	4.68%	5.26%	1.75%	12.87%	9.94%	6.43%	7.60%	8.77%	6.43%	14.62%	8.77%

Figure 7.18: Set of all vehicular components/assets are defined in the use case, with the rate of inferred threats, and the ratio between the number of the selected security requirements for each vehicular unit and the total number of all security requirements are stored in security requirements knowledge-base

The figure shows the number of inferred threats for each component/asset in our use case. The figure also illustrates the ratio between the selected security requirements and the total stored security requirements in the KB. This ratio shows the amount of data selected by the framework out of a significant number of existing security requirements to solve a particular security issue. Therefore, the framework studies and scans all the security requirements content and its relationships with the other security characteristics to select suitable ones. For example, the framework scans all the security requirements stored in the KB and selects 5.85% of the existing security requirements to protect the HMI Interface unit from different malicious activities. In order to reach this ratio by manual approach, we could expect consuming much time and unreasonable effects. We can conclude that the framework focuses on suggesting the applicable security requirements for a particular component/asset to the vehicular design life cycle. Outcomes could be managed and checked by the vehicular security architect to decide which could be selected or ignored.

The manual approach in this decision could be a complicated process regarding the complexity of the vehicular design, and the interaction among multiple components/assets within the vehicle network. This approach would not be the best way to manage security testing and check the proper security requirements for addressing security gaps. The following table 7.5 gives a comparison between the proposed framework and the manual approach for following the same activities as the framework does.

The table illustrates a comparison between the framework's activities and the same actions that could be done by manual way. It includes the comparison from multiple criteria that could generally perform the comparison between these two types. This comparison is defined as follows:

- **Completeness**: The diversity of security requirements contents and the volume of several

Table 7.5: Comparison between activities of the framework and manual approach in managing and handling security issues in the vehicle design

#	**Criteria**	Framework	Manual
1	Completeness	ensure completeness	limited to small number of items
2	Correctness	assure correctness	unreasonable effort
3	Adaptability	fully adaptable for different input forms	the adaptability is based on user expertise
4	Human Interaction	limited to checking the final results	need expert user aware of the vehicular components/assets, their security properties, security requirements, and potential threats
5	Equivalency	equivalent for simple and complex use cases	equivalent only for simple use cases
6	Mistakes	significant reduction in mistakes	prone to mistakes
7	Evaluation	objective	subjective
8	Time of Execution	faster, which takes less than one hour	time consuming, up to several days
9	Scalability	full-scalable can be applied to parallel and distributed environment	limited to number of available experts
10	Effectiveness	more effective, it gives very high accuracy results	less effective, gives low accuracy results

documents that need to be reviewed to determine a specific security requirement(s) to address a particular security issue, could not ensure the completeness. We argue that the manual approach cannot guarantee to find the best security requirement for an existing security risk because it is suitable for a small number of items. The framework scans all the components/assets of the vehicle and reviews all their vulnerabilities to match the best solution (security requirements) for existing security issues.

- **Correctness**: Manual approaches in finding a specific security solution for an issue could take much time, with unreasonable effort. The framework applies a set of logical inference rules based on the common security characteristics among components, threats, and security properties for selecting relevant security requirements for a particular security issue. This process is performed in a particular set of time with high accuracy of results.

- **Adaptability**: As mentioned before, the framework is developed to be fully-adaptable to manage multiple input forms. It could be not easy in a manual way because it is based on the user's expertise to be fully aware of cybersecurity detail in the vehicular design.

- **Human Interaction**: The framework needs less-human interaction; activities are performed automatically, then the security architect/engineer could check outcomes to decide which could be selected or ignored. However, manually, the complete process is based on the human interaction, which needs a group of experts who should be aware with the internal vehicular design and its security issues to check the correctness of the applied security requirements. It is also a crucial manual process to specify the exact new security requirements that can address the existing security issues.

- **Equivalency**: Equivalent with simple and complex examples in the automotive domain. The manual approach could be only possible for simple examples.
- **Mistakes**: The manual approach could still have a high rate of mistakes regarding human interactions, where the framework is fully managed by a set of rules that are applied automatically according to the ontology data input. We cannot claim that the framework provides a set of outcomes completely out of mistakes, but the framework has a significant reduction in mistakes than a manual approach.
- **Evaluation**: The manual approach is usually based on personnel opinion and decision, which means the evaluation is considered a subjective approach. As discussed earlier, this work uses the UNECE threats list, which is defined as a set of potential high-level threats and relevant security vulnerabilities in the automotive domain. Also, security requirements integrated within this work are based on the protection profile according to common criteria and security standards. Furthermore, the framework's outcomes are based on the results of multiple calculations on the input ontology model based on the knowledge base's data, which is strict adherence to specific criteria. Therefore, the framework's evaluation is objective because it is a decision based on facts according to the data that are selected from the specified resources. Objective and subjective evaluation are discussed in detail in [bsm19].
- **Time of Execution**: The manual approach could take more time than expected. According to Figure 7.18, we estimate the average time taken by the framework to completely infer the exact potential threats and deduce the relevant security requirements for all 18 vehicular components/assets in our case study. Figure 7.19 illustrates the average time taken for each component and asset to complete the security activities as discussed before. The figure illustrates two experiments are applied to the use case to evaluate the average time of the framework to complete the whole process. Table 7.6 illustrates some information about the number of components, security requirements, and potential threats are used within these experiments to determine the execution time.

  The figure shows the framework's overall time in two separate experiments based on the use case. The average time of these two experiments is 17.53 minutes. This estimation is based on the execution time on the complete use case. Table 7.6 gives more detail on the number of resources that are used within this process to infer a set of potential threats for each component and asset. The table also shows the total number of security requirements that are handled to select the most applicable set of security requirements for each component and assets against the deduced potential threats.

Table 7.6: Details about the number of resources are used in the case study, and the average time spent by the framework and by the manual approach to achieve this task

Resources	Number	Average Time (minutes)	
		Framework	Manual
Security Requirements	171	17.53	264*
Potential Threats	75		
Components/Assets	18		

The average estimated time of the manual approach is calculated according to the average

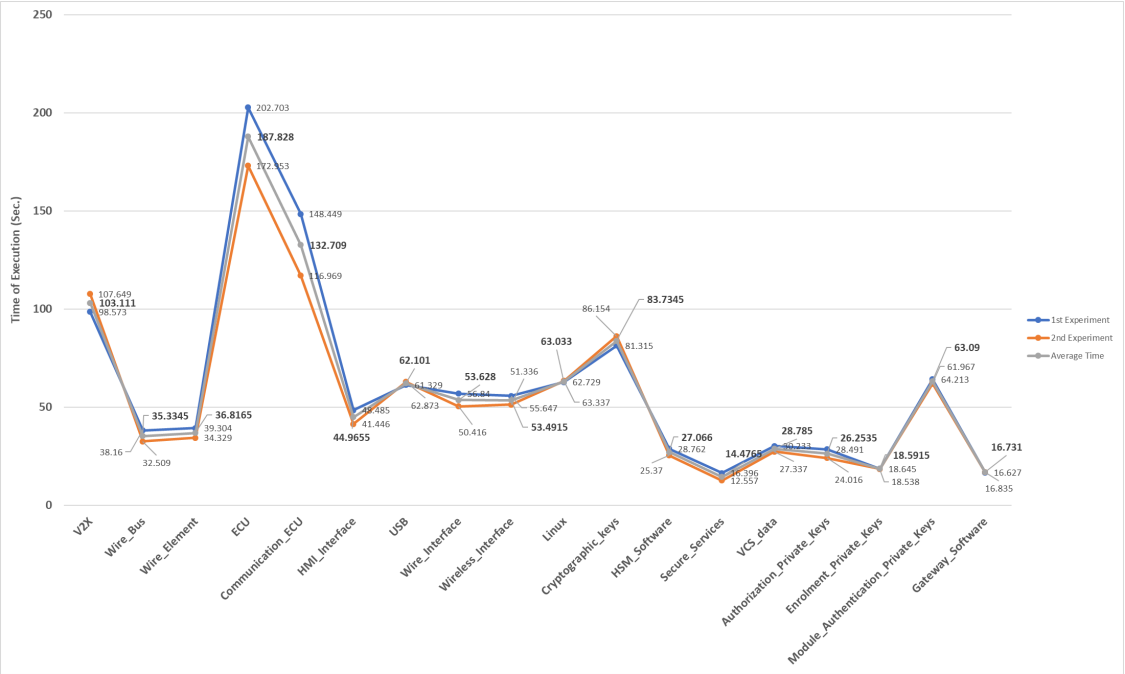


Figure 7.19: The average time between the two applied experiments to the use case

time that a person can read the context for each security requirement, details about the threats, and understand the security properties in the use case component/asset. If we say that, just one minute is sufficient for a person to check each resource in this use case, we can say that the average time to complete the process is 264* minutes. The '*' represent the variant in the elapsed time based on the human experience in this field. In this estimation of the manual approach we do not consider the time needed to perform threats analysis or prioritise an appropriate security requirement for a particular potential threat. Therefore, we can conclude that the framework is 15x faster than the manual approach.

- **Scalability**: The integration of the ontology approach within this work gives advantages of the framework to be scalable and applicable to other environments. In comparison, the manual approach will be limited according to human expertise.

- **Effectiveness**: According to the previously discussed comparison metrics, we can say that the framework is more effective for saving time and effort in handling the security issues in the vehicular domain. Also, it is considered more effective than the manual solution for giving high-quality results.

Note: All experiments are conducted within this work on intel Core i7 vPro 7th Generation 2.80GHz, with 32 GB RAMs.

7.5 Chapter Summary

The chapter used an automotive practical use case to assess the feasibility of the proposed framework. Four experimental examples are designed to represent various data input types of the

proposed framework. The chapter represents the outcomes of the framework for each experiment. It evaluates findings to show that the proposed framework can ensure the correctness of the security requirements implemented within the vehicle structure, and handle existing security deficiencies in vehicle design. Then the chapter provides a description of findings obtained and a comparison of the framework's activities with the manual way.

The next chapter represents a summary of the full thesis work and presents the impact of the research. The chapter ends with the main research limitations and suggests new future directions to cope with these limitations.

8 Summary and Future Work

This chapter summarizes the whole thesis work, which provides an overview of the research contribution and impact. It also includes the research limitations and addresses by our future direction to increase the proposed framework's capabilities and recommend new ideas to cope with the existing limitations.

8.1 Thesis Summary

Cybersecurity is one of the most challenging topics in the automotive domain. This is because of integrating Internet connectivity with modern vehicles to provide advanced features. The Internet connectivity within the vehicular network could also lead to multiple malicious attacks; such as device connects to the Internet. Cybersecurity in the automotive domain is protecting vehicular data and is responsible for protecting the critical vehicular units from different cyber attacks. Modern cars consist of a wide range of electronic and electrical units for managing and controlling functional safety. The attacker could compromise a critical vehicular unit such as the engine control unit or brake control unit, leading to unexpected negative consequences ranging from injuries to death.

A proposed ontology-based cybersecurity framework is introduced in this thesis work for the automotive domain. The framework aims to check the correctness of the applied security requirements within the vehicular design by applying a series of logical inference rules to ensure that the security requirements are fulfilled. Sequences of procedures are applied for each vehicular component/asset individually to verify and validate the correctness of the selected security requirements. Additionally, it can manage security gaps identified by the verification and validation process. This is done by suggesting a suitable set of security requirements that could be integrated within the vehicular design to protect the assets and components within the vehicular network. It evaluates the security level achieved after the security requirements are applied to the vehicle. Then it provides a new range of security requirements that could be applied to the vehicle to improve the level and meet the actual security goal.

The framework consists of multiple phases, such as data digestion, logical verification, security validation, gap analysis, and security enhancement. Data digestion is responsible for reading and scanning all input data in ontology form, representing all vehicle entities. This phase consists of activities to perform an in-depth analysis of the input data to obtain the necessary knowledge needed in the verification and validation process to validate vehicle security criteria against specific security vulnerabilities. The verification phase is based on a theorem proving method for verifying the correctness of the ontology structure's consistency and assuring the ontology hierarchy's completeness with the existing relationships. The verification uses logical methods to build complete, precise mathematical terms for giving accurate results. The framework checks multiple object property relationships between triples in the ontology structure to verify the security criteria among multiple entities.

The framework performs the validation process to determine whether the identified security

requirements are validated against the existing security issues. This phase is based on the testbed execution to perform a set of sequence procedures for each vehicular component/asset individually to evaluate the correctness of the selected security requirements. This testbed approach consists of a set of processes that are inspired by the ISO risk management process. The process consists of sub-processes (i.e., risk identification, risk analysis, risk evaluation, and risk treatment). These processes are adapted to fit in this work and create a new framework that is able to validate the selected security requirements against a wide range of potential threats, to protect a particular component/asset. Then the framework compares the outcomes of this process with the previously selected security requirements to completely validate that the selected security requirements are determined to address the security weaknesses and protect the chosen asset/component. In certain instances, some of the previous security requirements might not be 100% matched to the inferred ones. These selected security requirements, however, could be successful in resolving security issues and protecting vehicle components/assets. Therefore, the framework examines the correctness of these security requirements against the inferred potential threats. It analyzes the security characteristics of the selected security requirements and deduces a new collection of threats from the KB threats of the framework and then compares these threats to the threats previously inferred. Therefore, the framework is not acting only as a verifier and validator of the security requirements in the vehicular design, but also for addressing the existing security gaps to protect the vehicular assets/components from different malicious actions. Additionally, it manages these gaps to improve the overall security level to meet the actual security target.

The framework is designed to be fully adaptable to handle different forms of ontology inputs representing the vehicular components/assets interactions with the security protections and their relations with the applied security requirements. The input ontology model could contain a set of previously selected potential threats that have a negative impact on the existing vehicle units. Four experiments for the case study are created for evaluating the effectiveness of the proposed cybersecurity framework for checking and handling the existing security issues in vehicular design.

8.2 Impact of the Research

According to the discussed research outcomes, the framework studies all relationships among the vehicular entities in the ontology model (i.e., components, assets, potential threats, and relevant security requirements). The framework performs a logical verification and testbed execution to ensure that the applied security requirements for each component/asset are fulfilled. It ensures the correctness, completeness, and effectiveness of the applied security requirements that are selected to protect the vehicular units against the existing security issues. The discovered security issues can also be managed and handled by the framework. It can fill the existing security gaps with relevant security requirements. These requirements are considered a set of suggested or recommended security requirements, which assist the vehicular security architect in selecting and finding the most proper security requirements that help to fulfil the existing security gaps in the vehicular design. The framework gives initial estimations of the security target, and the security achieved level after the suggested security requirements are applied. These estimations are assumed as preliminary numbers to define the first indication of the achieved security level against the required security target level, which indicates how far the security target to be achieved.

According to the forehead discussed points, the proposed framework is considered a methodological framework for automating the mapping process for handling security vulnerabilities and relevant security requirements, as discussed in the methodological gap analysis in Section 1.2.

This automation process helps to investigate a massive number of ontology entities to check if all the applied security requirements address existing security issues. The estimated security target and security achieved values represent a preliminary expectation of the current security level after applying security requirements to the vehicular model, which helps estimate how secure a vehicle is. As discussed and presented in the experiment "Case 2: The ECU Component with Importing of Potential Threats", Section 7.3.2, the framework evaluates the ST and SA of the ECU unit. The outcomes of these examples illustrate that the SA values give an initial estimation about the ECU's current security level, according to the applied security requirements against the existing potential threats. These values show a simple mathematical estimation parameter representing the degree of security of a vehicle or vehicular components/asset. Consequently, this framework can addresses the research gap **P1** (i.e., vehicle manufacturers check how secure their cars are).

According to automating security requirements verification and validation activities, as discussed in Chapter 6, the framework can handle and manage multiple ontology entities for assuring correctness and completeness of the applied security requirements against existing potential threats. These activities aim to audit and discover security gaps, ineffective, or redundant security requirements. As discussed in "Case 1: The asset "VCS data" with the Integration of Potential Threats and Security Requirements", Section 7.3.1, the outcomes of the framework are matched with the security requirements and potential threats that are integrated with this ontology example. However, in the "Case 3: The V2X Unit with Importing of Relevant Security Requirements" in Section 7.3.3, the only security requirements are integrated within the input example. The framework checks the correctness of the existing security requirements against identified potential threats (i.e., according to the risk analysis process) by deducing a new set of security issues from these security requirements. Only one threat is not addressed because no common properties are detected between the existing security requirements and the identified threats. Similar to follows in the other cases in Section 7.3.2 and Section 7.3.4, the outcomes demonstrate that the proposed framework can perform V&V for multiple input forms and handle it under multiple circumstances. Therefore, this framework is thought of as a methodological approach to address the **P2** (i.e., challenging in security requirements verification and validation processes).

The integration of the ontology approach within this research context makes the proposed cybersecurity framework completely scalable and versatile in managing a wide range of security requirements. As presented in Chapter 4 security requirements from the IEC 62443-4-2 and the V2X HSM protection profile are represented in an ontology form and used in this work. The framework follows an approach for building and understating relationships among multiple ontology entities to give more accurate results. This approach is developed to automatically map security issues with an appropriate set of security requirements to address these issues. The outcomes of the case study, as presented in Section 7.4 illustrate that the framework handles these both security requirement types (i.e., IEC 62443 and protection profile) and map each vehicular component/asset with relevant security requirements. Each vehicular element (i.e., component/asset) has a set of security requirements suggested by the framework to protect it from different potential threats. The selected security requirements are chosen among 171 used in this work are collected from different sources. Therefore, the proposed cybersecurity framework is considered the desirable approach for addressing the research gap **P3** (i.e., integrating relevant security requirements).

8.3 Research Limitations and Future Directions

We focus on two main limitations within this research context. Our future work aims to cope with these limitations by integrating new ideas and approaches, which could be suitable for our research endeavours.

8.3.1 Machine Learning Approach

In order to add new security requirements, we still need someone who has enough level of expertise in automotive cybersecurity. This person needs to understand the contents of security requirements and extract specific features that are used as common security measures to build a security requirements knowledge-base. This process is conducted in this work in a completely manual way, which takes plenty of time and effort to achieve. In addition, it is a quite challenging process for building an ontology model of a vehicular representation design, setting all relationships among different entities (i.e., components, assets, security measures, vulnerabilities, severity degrees, and security levels). As mentioned before, we define all the information we need to build the ontology model in a spreadsheet, and then we use Cellfire to import these data into OWL ontologies. However, this process still needs to trace all the axioms in the ontology and define a proper set of expressions to import all the previous data into an ontological form. We propose in the future to use a machine learning approach which can automatically perform text analysis based on semantic similarities approach to extract features from a text that represent the security properties of a particular security requirement. Then the extracted data is automatically integrated with the ontology knowledge base.

8.3.2 Adaptability of the Framework for Relevant Domains

The current version of the cybersecurity framework is proposed for the automotive domain. The framework can verify and validate security requirements against the potential threats in different transportation domains such as aviation, railways, and others in case the complete details are available. The integration of the IEC 62443 standard in the current research work allows the framework to be applied for multiple cyber-physical systems - CPS and Internet of things - IoT applications. However, to expand the proposed framework's functionality to be combined with other application domains, we need to identify new threats catalogues for these domains to be matched entirely with the required research topic. In addition, it is necessary to study and understand the relationships between multiple nodes in other domains so we can construct a complete ontology model to determine the security requirements according to the current security issues and to verify and validate the applied security requirements' correctness in order to build secure applications. As there are many different kinds of cyberattacks and threats in many different fields, future research is expected to include updating our current knowledge-base and developing new threats and relevant security requirements to be combined into other research areas.

9 Thesis Statement

Cybersecurity is considered an essential part of any system engineering development lifecycle. It is the practice of protecting data, networks, and devices from different cyber incidents. Existing security vulnerabilities in a system give attackers a way to reach malicious targets, such as collecting or destroying critical data, stealing credit card information, passwords, images, etc. In the automotive domain, the situation is different. Cybersecurity is protecting sensitive data and protecting critical vehicular units controlling functional safety in a vehicle. A cyberattack scenario on a vehicle on the highway could lead to unexpected negative consequences ranging from injuries to death.

Furthermore, this work introduces an ontology-based cybersecurity management framework for security issues in the automotive domain. It automates building relationships to define a complete tracking path from threats to security requirements to address security vulnerabilities in the vehicular design, which is not defined explicitly. These relationships also enhance outcomes of the security verification process for verifying the logical correctness of security requirements. However, security requirements could be selected and applied to the vehicular design without a clear understanding of security issues. These requirements could be redundant or not applicable for addressing the real existing security issues. Therefore, the framework provides a methodological approach for validating the applied security requirements based on the ISO risk management process. The framework performs a reverse action based on the identified relationships among multiple entities in the ontology model to deduce a new set of potential threats according to security vulnerabilities are identified in the input model. Then it checks the effectiveness of selected requirements against these threats.

The framework is developed to be fully-adaptable for handling different forms of ontology inputs representing relationships of vehicular elements with security issues and the applied security requirements. Additionally, it intends to manage the identified security gaps by suggesting a suitable set of security requirements. The security architect could check these requirements to decide which are suitable for the system design.

We investigate the performance of the proposed cybersecurity framework by using a real automotive case study. According to outcomes of the conducted experiments on the case study, we conclude that the proposed framework's performance is 15x times faster with higher accuracy and efficiency than a manual way, performed by highly expert personnel.

10 Publications

Over five years of work in this research, multiple outcomes have been documented and disseminated in various official events such as journals, book chapter, conferences and workshops.

10.1 Journal Papers

JDI Journal - (Published): Shaaban, Abdelkader Magdy, Christoph Schmittner, Thomas Gruber, A. Baith Mohamed, Gerald Quirchmayr, And Erich Schikuta. "Automated Ontology-Based Security Requirements Identification for The Vehicular Domain." Journal of Data Intelligence 1, No. 4 (2020): 401-418.
Ranking: C
Many electronic and electrical systems are now incorporated with modern vehicles to control functional safety. Lack of security protection mechanisms in vehicular design may lead to different ways of executing malicious attacks against the vehicular network. These attacks may have various types of negative consequences, such as safe vehicle operation. This work presents an ontology-based framework as a new automated approach to verify and validate security requirements against security issues in the vehicular domain. The system also applies a set of logical rules to identify a set of security requirements as a category of necessary security requirements that could be proposed to be integrated within the vehicle design to address a specific security issue.

IARIA Journal - (Published): Schmittner, Christoph, Martin Latzenhofer, Abdelkader Magdy Shaaban, Arndt Bonitz, and Markus Hofer. 2019. "Towards a Comprehensive Automotive Cybersecurity Reference Architecture." 12(1 & 2):1–12.
Ranking: C
This paper analyzes the technological and legal state of the art of automated driving for smart urban mobility. We conclude that the current state of the art is not yet sufficient with the complex requirements of such an environment. We identified four current challenges to a comprehensive traffic road system: The interoperability of the components among the vehicles as well as the infrastructure elements, connectivity and communication tasks especially for interacting and cooperation of the different components, Intelligent Transport System (ICT) in general and cybersecurity issues to address security threats, and privacy finally aspects which subsume protection requirements of personal data of the vehicle drivers. There are efforts to form a compliant legal and technological framework, but all these considerations are not yet completed.

10.2 Book Chapter

Book Contribution - (Accepted): Shaaban, Abdelkader Magdy, Christoph Schmittner, Thomas Gruber, A. Baith Mohamed, Gerald Quirchmayr, and Erich Schikuta. Ontology-Based Security Requirements Framework for Current and Future Vehicles. Vol. Data Science and Big Data Analytics in Smart Environments. CRC Press Taylor & Francis Group.

Ranking: B
Vehicular engineering process aims to develop a secure and safe vehicle with a high level of safety-security assurance. Safety and security are considered two sides of the identical coin. Therefore, improving security in automotive manufacturing is necessary to protect the vehicle from various attack scenarios that threaten safety. Consequently, cybersecurity requires to be a part of the development phases of the vehicular industry. The significant challenge is how to manage hundreds or thousands of vehicular data related to all components, threats, vulnerabilities, and protection profiles. The traditional security verification and validation approaches could miss some of the security flaws, which leads to a threat to the whole vehicle. This work aims to introduce a vehicular security verification and validation model. The model is an ontology-based approach that aims to create a knowledge representation of the vehicular components, assets, threats, and others with all related data. Then it performs verification and validation to determine whether or not the security requirements met under the actual security conditions. Additionally, the model improves the security level of the vehicle by choosing further security requirements.

10.3 Conferences

IDIMT Conference - (Published): Abdelkader Magdy, Shaaban, and Schmittner Christoph. 2020. "Threatget: New Approach Towards Automotive Security-By-Design." Pp. 413–19 In IDIMT-2020 Digitalized Economy, Society and Information Management. Kutná Hora, Czech Republic.
Ranking: C
Cybersecurity has become one of the biggest challenges in the automotive engineering process. Identifying the exact potential threats that can affect the vehicle system in the early stages of the vehicle life cycle is necessary because, once the vehicle is designed, it will be difficult to add security. This paper introduces ThreatGet as a novel threat modeling methodology, specifically developed for the automotive sector. ThreatGet can be integrated with the early stages of vehicle system design to identify threats and evaluate the associated risk to measure the overall risk and define security goals.

iiWAS Conference - (Published): Shaaban, Abdelkader Magdy, Christoph Schmittner, Thomas Gruber, A. Baith Mohamed, Gerald Quirchmayr, and Erich Schikuta. 2019. "Ontology-Based Model for Automotive Security Verification and Validation." Pp. 73–82 in Proceedings of the 21st International Conference on Information Integration and Web-Based Applications &; Services, iiWAS2019. New York, NY, USA: Association for Computing Machinery
Ranking: B
Modern automobiles are considered semi-autonomous vehicles regarding new adaptive technologies. New cars consist of a vast number of electronic units for managing and controlling the functional safety in a vehicle. In the vehicular industry, safety and security are considered two sides for the same coin. Therefore, improving functional safety in the vehicular industry is essential to protect the vehicle from different attack scenarios. This work introduces an ontology-based model for security verification and validation in the vehicular domain. The model performs a series of logical quires and inference rules to ensure that the security requirements are fulfilled. It endeavors to enhance the current security state of a vehicle by selecting additional security requirements that can handle existence security weaknesses and meet the actual security goal.

ICONIP Conference - (Published): **Shaaban, Abdelkader Magdy, Christoph Schmittner, Gerald Quirchmayr, A. Baith Mohamed, Thomas Gruber, and Erich Schikuta. 2019. "Toward the Ontology-Based Security Verification and Validation Model for the Vehicular Domain." Pp. 521–529 in Neural Information Processing, edited by T. Gedeon, K. W. Wong, and M. Lee. Cham: Springer International Publishing.**
Ranking: A
Security verification and validation is an essential part of the development phase in current and future vehicles. It is essential to ensure that a sufficient level of security is achieved. This process determines whether or not all security issues are covered and confirms that security requirements and implemented measures meet the security needs. This work proposes a novel ontology-based security verification and validation model in the vehicular area. Ontologies allow creating a comprehensive view of threats and security requirements. The proposed model performs a series of queries and inference rules to the comprehensive view to ensure the compliance of vehicle components with security requirements.

RSSRail Conference - (Published): **Schmittner, Christoph, Peter Tummeltshammer, David Hofbauer, Abdelkader Magdy Shaaban, Michael Meidlinger, Markus Tauber, Arndt Bonitz, Reinhard Hametner, and Manuela Brandstetter. 2019. "Threat Modeling in the Railway Domain." Pp. 261–71 in Reliability, Safety, and Security of Railway Systems. Modelling, Analysis, Verification, and Certification. Vol. 11495, Lecture Notes in Computer Science, edited by S. Collart-Dutilleul, T. Lecomte, and A. Romanovsky. Cham: Springer International Publishing.**
Ranking: C
Connected and intelligent railway technologies like the Euro-pean Rail Traffic Management System (ERTMS) introduce new risks in cybersecurity. Threat modeling is a building block in security engineering that identifies potential threats in order to define corresponding mitigation. In this paper, we show how to conduct threat modeling for railway security analysis during a development life cycle based on IEC 62443. We propose a practical and efficient approach to threat modeling, extending existing tool support and demonstrating its applicability and feasibility.

VEHICULAR Conference - (Published): **Shaaban, Abdelkader Magdy, Christoph Schmittner, and Arndt Bonitz. 2019. "The Design of a Divide-and-Conquer Security Framework for Autonomous Vehicles." Pp. 94–102 in The Eighth International Conference on Advances in Vehicular Systems, Technologies and Applications. Rome, Italy.**
Ranking: C
The vehicular security engineering process endeavors to build up a secure vehicle with a high level of security assurance. The well-identified security flaws and conforming security countermeasures help to deliver secure vehicles. This work presents a newly Divide-and-Conquer security framework which can be integrated with the early stages of the vehicular development process to emphasize the security-by-design. The framework proposes to divide the vehicle components into separate layers and sublayers, according to common security parameters. Subsequently, the framework applies a series of security management actions to define potential threats and security vulnerabilities in a vehicle; thereupon, it selects a list of security countermeasures which can mitigate the vehicular's risk. Eventually, the framework performs a security verification and validation to ensure that the vehicle has been developed according to the highest degree of protection level.

IDIMT Conference - (Published): Shaaban, Abdelkader Magdy, Christoph Schmittner, and Thomas Gruber. 2019. "Tackling the Challenges of IoT Security Testing Using Ontologies." Pp. 411–18 In 27th Interdisciplinary Information Management Talks. Kutná Hora, Czech Republic.
Ranking: C
Cybersecurity needs to be an integral part of the development phases of the Internet of Things (IoT) technology. Heterogeneity of elements and diversity of the communication protocols generate new security issues which need to be addressed by proper security requirements. This work introduces an ontology-based security testing framework for IoT applications. The framework describes threats and related security requirements of an IoT application in ontologies form. These ontologies are used to validate and verify the security requirements against the threats to ensure that security requirements are fulfilled.

VEHICULAR Conference - (Published): Schmittner, Christoph, Martin Latzenhofer, Abdelkader Magdy Shaaban, and Markus Hofer. 2018. "A Proposal for a Comprehensive Automotive Cybersecurity Reference Architecture." Pp. 30–36 in The Seventh International Conference on Advances in Vehicular Systems, Technologies and Applications. Venice, Italy.
Ranking: C
Interconnection, complexity and software-dependency are prerequisites for automated driving and increase cybersecurity risks for the whole transportation system. Hence, information and communication technology infrastructure becomes a second layer for critical transportation infrastructure. In a recently started research project, we identify involved stakeholders and risks in a structured manner to integrate the diverging interests and objectives of authorities, road infrastructure providers and transport facilitators, which cannot even exclusively leave to the original equipment manufacturers. Based on the emerging risk scenarios, we develop a comprehensive architectural reference framework. Only if all components in the ICT infrastructure provide their services in a sufficient quality according to ensured security requirements, society can rely on the reliable automotive system

iiWAS Conference - (Published): Shaaban, Abdelkader Magdy, Christoph Schmittner, Thomas Gruber, A. Baith Mohamed, Gerald Quirchmayr, and Erich Schikuta. 2018. "CloudWoT - A Reference Model for Knowledge-Based IoT Solutions." Pp. 272–281 in Proceedings of the 20th International Conference on Information Integration and Web-Based Applications &; Services, iiWAS2018. New York, NY, USA: Association for Computing Machinery.
Ranking: B
Internet technology has changed how people work, live, communicate, learn and entertain. The internet adoption is rising rapidly, thus creating a new industrial revolution named "Industry 4.0". Industry 4.0 is the use of automation and data transfer in manufacturing technologies. It fosters several technological concepts, one of these is the Internet of Things (IoT). IoT technology is based on a big network of machines, objects, or people called "things" interacting together to achieve a common goal. These things are continuously generating vast amounts of data. Data understanding, processing, securing and storing are significant challenges in the IoT technology which restricts its development. This paper presents a new reference IoT model for future smart IoT solutions called Cloud Web of Things (CloudWoT). CloudWoT aims to overcome these limitations by combining IoT with edge computing, semantic web, and cloud computing. Additionally, this work is concerned with the security issues which threatens data in IoT application domains.

iiWAS Conference - (Published): Magdy Abdelkader, A. Baith Mohamed, Gerald Quirchmayr, and Erich Schikuta. 2017. "Towards a Security and Privacy Protection Model for Semantic Query Engines." Pp. 198–207 in Proceedings of the 19th International Conference on Information Integration and Web-Based Applications &; Services, iiWAS '17. New York, NY, USA: Association for Computing Machinery.
Ranking: B
The semantic web aims to describe information in terms of well-defined vocabularies and comprehends both data and knowledge to cope with meaning of data. Advanced search engines are used to retrieve precise information out of these knowledge resources. The main challenge is not only retrieving data but also how to keep data safe and protected against any form of attacks. In this paper, we propose a security aware based model for semantic search engines. Our work aims to combine advances in information technology, such as cloud technology, while addressing security issues which threaten the integrity of information. In particular security gaps and countermeasures of the semantic web are identified. ISO/IEC security requirements for the protection of personally identifiable information (PII) are presented to cover security vulnerabilities of the proposed model. Finally, the feasibility of our proposed model is checked against the N2Sky use case, a multi-cloud knowledge information management system for the computational intelligence community.

10.4 Workshops

AI4SE Workshop - (Published): Schmittner, Christoph, Abdelkader Magdy Shaaban, and Johannes Hellrich. 2019. "Automating the Construction of a Security Threat and Mitigation Pattern Library." in INCOSE Artificial Intelligence for Systems Engineering.
Ranking: C
Risk management is one of the most important aspects of the engineering process of secure systems. Here the challenge is to select a sufficient set of measures for risk treatment. Especially for information and cyber security risks deciding on a sufficient set of treatment measures or controls requires expert knowledge. Controls can cover multiple risks and can depend on other controls. We present here a concept how the knowledge of approved Protection Profiles can be processed with natural language processing (NLP) to form security patterns.

SafeComp Workshop - (Published): Shaaban, Abdelkader Magdy, Erwin Kristen, and Christoph Schmittner. 2018. "Application of IEC 62443 for IoT Components." Pp. 214–23 in Computer Safety, Reliability, and Security, Lecture Notes in Computer Science, edited by B. Gallina, A. Skavhaug, E. Schoitsch, and F. Bitsch. Cham: Springer International Publishing.
Ranking: C
Internet technology has changed how people live, work, connect and learn. It connects machines, devices, sensors, and people and enables communication. This enabled a revolution in the industrial perspective, which is named "Industry 4.0." Industry 4.0 is the application of automation and data exchange in manufacturing technologies. This rapid progression of industrial systems towards internet based production networks needs a flexible framework that facilitates addressing current and future vulnerabilities in Industrial Automation Control Systems (IACS). IEC 62443 series provides a standard methodology for building a secure infrastructure, which adapts the security requirements needed by IACS. The basic approach defined in the standard is to break down the system components into zones and conduits based on required security levels. This paper reuses this idea on a small scale to show how the same concept can be used to define zones

and conduits between mixed-criticality IoT components to improve the security on component level. The MORETO tool, which is currently under development by AIT, supports the security risk analysis process.

EuroS&PW Workshop - (Published): Ma, Zhendong, Aleksandar Hudic, Abdelkader Shaaban, and Sandor Plosz. "Security viewpoint in a reference architecture model for cyber-physical production systems." In 2017 IEEE European symposium on security and privacy workshops (EuroS&PW), pp. 153-159. IEEE, 2017.
Ranking: C
Cyber-physical Production Systems (CPPS) are one of the technical driving forces behind the transformation of industrial production towards "digital factory of the future" in the context of Industry 4.0. Security is a major concern for such systems as they become more intelligent, interconnected, and coupled with physical devices. For various security activities from security analysis to designing security controls and architecture, a systematic and structured view and presentation of security-related information is required. Based on the draft standard of Reference Architecture Model for Industry 4.0 (RAMI 4.0), we propose a practical approach to establish a security viewpoint in the CPPS reference architecture model. We investigate the feasibility of using an architecture modeling tool to implement the concept and leverage existing work on models of layered architecture. We demonstrate the applicability for security analysis in two example case studies.

10.5 Other Publications in 2016 & 2017:

During my Ph.D. studies, I have some other works related to ontology approaches. The ontology is proposed to be used as a new concept of providing solution specific problem over the cloud. Here is a list of publications have published in this area.

IWANN Conference - (Published): Schikuta, Erich, Abdelkader Magdy, Irfan Ul Haq, A. Baith Mohamed, Benedikt Pittl, and Werner Mach. 2017. "Searching the Sky for Neural Networks." Pp. 167–178 in Advances in Computational Intelligence, edited by I. Rojas, G. Joya, and A. Catala. Cham: Springer International Publishing.
Ranking: B
Sky computing is a new computing paradigm leveraging resources of multiple Cloud providers to create a large-scale distributed infrastructure. N2Sky is a research initiative promising a framework for the utilization of Neural Networks as services across many Clouds integrating into a Sky. This involves a number of challenges ranging from the provision, discovery and utilization of N2Sky services to the management, monitoring, metering and accounting of the N2Sky infrastructure. This paper focuses on the semantic discovery of N2Sky services through a human-centered querying mechanism termed as N2Query. N2Query allows N2Sky users to specify their problem statement as natural language queries. In response to the natural language queries, it delivers a list of ranked neural network services to the user as a solution to their stated problem. The search algorithm of N2Query is based on the semantic mapping of ontologies referring to problem and solution domains.

ICONIP Conference - (Published): Schikuta, Erich, Abdelkader Magdy, and A. Baith Mohamed. 2016. "A Framework for Ontology Based Management of Neural Network as a Service." Pp. 236–243 in Neural Information Processing, edited by A. Hirose, S. Ozawa, K. Doya, K. Ikeda, M. Lee, and D. Liu. Cham: Springer International Publishing.
Ranking: A
Neural networks proved extremely feasible for problems which are hard to solve by conventional computational algorithms due to excessive computational demand, as NP-hard problems, or even lack of a deterministic solution approach. In this paper we present a management framework for neural network objects based on ontology knowledge for the cloud-based neural network simulator N2Sky, which delivers neural network resources as a service on a world-wide basis. Core of this framework is the Neural Network Query Engine, N2Query, which allows users to specify their problem statements in form of natural language queries. It delivers a list of ranked N2Sky resources in return, providing solutions to these problems. The search algorithm applies a mapping process between a domain specific problem ontology and solution ontology.

The following chart illustrates a summary of the all discussed publications, ranks ranging from A, B, and C.

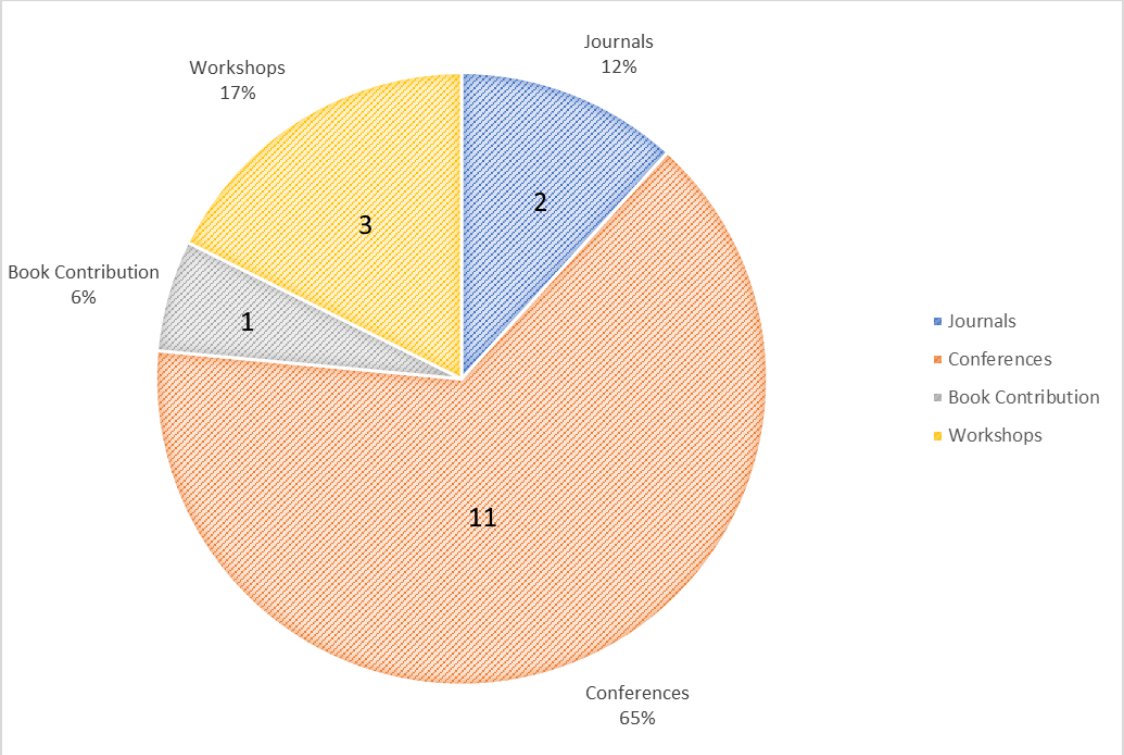


Figure 10.1: Summary of the publications

The figure illustrates that 65% of the publications were published in conferences, while 12% were published in journals. These are both percentages from a total of 17 publications published throughout my PhD, as previously discussed. Some works were published in workshops and other research findings as published within a book contribution.

Bibliography

[AA14] Abd El-Aziz Ahmed and Kannan Arputharaj. Literature review on xml security and access control to xml documents. Studies in System Science, 01 2014.

[AAA19] AAA. Automatic emergency braking with pedestrian detection. Technical report, American Automobile Association (AAA), Florida, United States, October 2019.

[AAL16] Hira Asghar, Zahid Anwar, and Khalid Latif. A deliberately insecure rdf-based semantic web application framework for teaching sparql/sparul injection attacks and defense mechanisms. Computers & Security, 58:63–82, 2016.

[ADM99] NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION. Federal motor vehicle safety standards and regulations. `https://one.nhtsa.gov/cars/rules/import/FMVSS/index.html`, 1999. (Accessed on: November 24, 2020).

[Adm16] NHTSA National Highway Traffic Safety Administration. Assessment of safety standards for automotive electronic control systems. Technical report, United States. National Highway Traffic Safety Administration, 2016.

[AdMK17] Jeremy Avigad, Leonardo de Moura, and Soonho Kong. Theorem proving in lean. `https://leanprover.github.io/tutorial/tutorial.pdf`, 2017.

[ADSW03] Christopher Alberts, Audrey Dorofee, James Stevens, and Carol Woody. Introduction to the octave approach. Technical report, Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst, 2003.

[AGK15] Mohamed Abomhara, Martin Gerdes, and Geir M Køien. A STRIDE-based threat model for telehealth systems. NISK Journal, pages 82–96, 2015.

[AIT19] AIT Austrian Institute of Technology, and LieberLieber Software. Why threatget? `https://www.ait.ac.at/fileadmin/mc/digital_safety_security/downloads/LieberLieber_Threatget_EN.pdf`, 2019. (Accessed on: June 12, 2020).

[AJH19] The VERGE Andrew J. Hawkins. Cars with high-tech safety systems are still really bad at not running people over. `https://www.theverge.com/2019/10/4/20898773/aaa-study-automatic-emergency-braking-pedestrian-detection`, 2019. (Accessed on: January 23, 2020).

[Alh17] Alabbas Alhaj. ISO 26262 functional safety standard and the impact in software lifecycle. `http://rgdoi.net/10.13140/RG.2.2.12486.16963`, 2017. JOURNAL OF UNIVERSITY OF APPLIED SCIENCES.

[AMC20] Shaaban Abdelkader Magdy and Schmittner Christoph. THREATGET: NEW APPROACH TOWARDS AUTOMOTIVE SECURITY-BY-DESIGN. In IDIMT-2020 Digitalized Economy, Society and Information Management, pages 413–419, 2020.

[AMS19] Arndt Bonitz Abdelkader Magdy Shaaban, Christoph Schmittner. The design of a divide-and-conquer security framework for autonomous vehicles. In The Eighth International Conference on Advances in Vehicular Systems, Technologies and Applications, pages 49–102, 2019.

[APM11] Yudistira Asnar, Elda Paja, and John Mylopoulos. Modeling design patterns with description logics: A case study. In International Conference on Advanced Information Systems Engineering, pages 169–183. Springer, 2011.

[Arc20] Enterprise Architect. Enterprise architect by sparx systems. https://www.sparxsystems.eu/start/home/, 2020. (Accessed on: October 6, 2020).

[AS12] N AMSC and A AREA SAFT. Department of defense standard practice system safety, 2012. https://www.dau.edu/cop/armyesoh/DAU%20Sponsored%20Documents/MIL-STD-882E.pdf.

[Ass13] Oracle Assetidentification. Overview to asset identification. https://docs.oracle.com/cd/E26228_01/doc.93/e21539/ovrvw_asset_id.htm#WEAFA124, 2013. (Accessed on: June 25, 2020).

[AUT03] AUTOSAR. Autosar - automotive open system architecture. https://www.autosar.org/, 2003. (Accessed on: November 24, 2020).

[AVL20] AVL. Automotive cybersecurity - a holistic approach to the protection of vehicles. https://www.avl.com/web/guest/services1/-/asset_publisher/gYjUpY19vEA8/content/automotive-cyber-security, 2020. (Accessed on: October 10, 2020).

[BBG17] P. Bagade, A. Banerjee, and S.K.S. Gupta. Chapter 12 - validation, verification, and formal methods for cyber-physical systems. In Houbing Song, Danda B. Rawat, Sabina Jeschke, and Christian Brecher, editors, Cyber-Physical Systems, Intelligent Data-Centric Systems, pages 175 – 191. Academic Press, Boston, 2017.

[BDLJ15] Jasper Bogaerts, Maarten Decat, Bert Lagaisse, and Wouter Joosen. Entity-based access control: supporting more expressive access control policies. In Proceedings of the 31st Annual Computer Security Applications Conference, pages 291–300. ACM, 2015.

[BHS03] Valerie Bönström, Annika Hinze, and Heinz Schweppe. Storing rdf as a graph. In Web Congress, 2003. Proceedings. First Latin American, pages 27–36. IEEE, 2003.

[BLV+08] C. Blanco, J. Lasheras, R. Valencia-García, E. Fernández-Medina, A. Toval, and M. Piattini. A systematic review and comparison of security ontologies. In 2008 Third International Conference on Availability, Reliability and Security, pages 813–820, March 2008.

[Bra14] Jens Braband. Towards an IT Security Framework for Railway Automation. In Embedded real-time software and systems (ERTS^2 2014), TOULOUSE, France, February 2014.

[Bru99] Schneier Bruce. Attack trees. https://www.schneier.com/academic/archives/1999/12/attack_trees.html, 1999. (Accessed on: October 3, 2020).

[bRW19] Safryn by Richard Wood. What's the difference between qualitative and quantitative risk analysis? https://www.safran.com/blog/whats-the-difference-between-qualitative-and-quantitative-risk-analysis, 2019. (Accessed on: August 8, 2020).

[bsm19] bsmimpact. Objective vs subjective. https://bsmimpact.com/objective-vs-subjective/#:~:text=Based%20on%20or%20influenced%20by%20personal%20feelings%2C%20tastes%2C%20or%20opinions.&text=Objective%3A,in%20considering%20and%20representing%20facts, 2019. (Accessed January 18, 2021).

[Bus20] Business Research Methodology. Action research. https://research-methodology.net/research-methods/action-research/, 2020.

[BW15] Michele Bertoncello and Dominik Wee. Ten ways autonomous driving could redefine the automotive world. McKinsey & Company, 6, 2015.

[CAFC+16] Samarjit Chakraborty, Mohammad Abdullah Al Faruque, Wanli Chang, Dip Goswami, Marilyn Wolf, and Qi Zhu. Automotive cyber–physical systems: A tutorial introduction. IEEE Design & Test, 33(4):92–108, 2016.

[Car17] Carrie Cox and Andrew Hart. How autonomous vehicles could relieve or worsen traffic congestion. Technical report, Here Technologies, 2017.

[Car18] Car 2 Car Communication Consortium. Protection Profile V2X Hardware Security Module. https://www.car-2-car.org/fileadmin/documents/Basic_System_Profile/Release_1.3.0/C2CCC_PP_2056_HSM.pdf, 2018.

[Car19] Car 2 Car Communication Consortium. Protection Profile V2X Hardware Security Module. https://www.car-2-car.org/fileadmin/documents/Basic_System_Profile/Release_1.4.0/C2CCC_PP_2056_HSM.pdf, 2019.

[CC10] Chidchanok Choksuchat and Chantana Chantrapornchai. Benchmarking query complexity between rdb and owl. In International Conference on Future Generation Information Technology, pages 352–364. Springer, 2010.

[Cen16] Centro Criptológico Nacional. Protection Profilefor Trusted Platform for secure communications. EAL2+. Protection profile, CCentro Criptológico Nacional, 2016.

[Cha10] Threats and Countermeasures for Web Services. https://msdn.microsoft.com/en-us/library/ff650168.aspx#MessageEncryption, 2010. (Accessed on: August 9, 2017).

[Com16] European Commission. Car industry: European commission tightens rules for safer and cleaner cars. https://ec.europa.eu/commission/presscorner/detail/fr/MEMO_16_168, 2016. (Accessed on: June 11, 2020).

[Com20] Common Vulnerabilities and Exposures. CVE list home. https://cve.mitre.org/cve/, 2020. (Accessed on: October 27, 2020).

[Cri17] Common Criteria. Common methodology for information technology security evaluation. https://www.commoncriteriaportal.org/files/ccfiles/CEMV3.1R5.pdf, 2017.

[CVE18] CVE. Vulnerability details : CVE-2008-4306. https://www.cvedetails.com/cve/CVE-2008-4306/, 2018. (Accessed on: October 27, 2020).

[CVWF] R. Craft, R. Vandewart, G. Wyss, and D. Funkhouser. An open framework for risk management, article. https://digital.library.unt.edu/ark:/67531/metadc702593/. (Accessed on: January 3, 2021).

[dAe15] Verband der Automobilindustrie eV. Automation: From driver assistance systems to automated driving. VDA Magazine-Automation, page 2, 2015.

[emb18] embitel. Understanding how iso 26262 ASIL is determined for automotive applications. https://www.embitel.com/blog/embedded-blog/understanding-how-iso-26262-asil-is-determined-for-automotive-applications, 2018. (Accessed on: September 9, 2020).

[emb19] embitel. How HARA Helps Functional Safety (ISO 26262) Consultants to Determine ASIL Values and Formulate Safety Goals. https://www.embitel.com/blog/embedded-blog/hara-by-iso-26262-standard-for-your-functional-safety-project, 2019. (Accessed January 25, 2020).

[ene20] SyC Smart energy. IEC 62443. https://syc-se.iec.ch/deliveries/cybersecurity-guidelines/security-standards-and-best-practices/iec-62443/, 2020. (Accessed on: July 5, 2020).

[ESSK19] Magdy El Sadany, Christoph Schmittner, and Wolfgang Kastner. Assuring compliance with protection profiles with threatget. In International Conference on Computer Safety, Reliability, and Security, pages 62–73. Springer, 2019.

[EVI08] EVITA. E-safety vehicle intrusion protected application - FINAL DRAFT. https://trimis.ec.europa.eu/sites/default/files/project/documents/20130702_175923_78998_EVITA_ProjectSummary.pdf, 2008.

[FGH+10] Benjamin Fabian, Seda Gürses, Maritta Heisel, Thomas Santen, and Holger Schmidt. A comparison of security requirements engineering methods. Requirements Engineering, 15(1):7–40, 2010.

[Fin17] Fine-Grained Access Control for RDF Data. https://docs.oracle.com/cd/E11882_01/appdev.112/e25609/fine_grained_acc.htm#RDFRM99941, 2017. (Accessed on: July 17, 2017).

[FIR20] FIRST. Common vulnerability scoring system version 3.1: Specification document. `https://www.first.org/cvss/specification-document`, 2020. (Accessed on: October 1, 2020).

[GB11] L. Gallon and J. J. Bascou. Using cvss in attack graphs. In 2011 Sixth International Conference on Availability, Reliability and Security, pages 59–66, 2011.

[GCM] Meysam Ghanavati, Animesh Chakravarthy, and Prathyush P. Menon. Analysis of automotive cyber-attacks on highways using partial differential equation models. 5(4):1775–1786.

[GGV] Benjamin Glas, Jens Gramm, and Priyamvadha Vembar. Towards an information security framework for the automotive domain. page 16.

[GGV15] Benjamin Glas, Jens Gramm, and Priyamvadha Vembar. Towards an information security framework for the automotive domain. Automotive-Safety & Security 2014, 2015.

[Gra08] Justin Gray. A framework for the automated alignment of ontologies. 2008.

[GSM+12] Dip Goswami, Reinhard Schneider, Alejandro Masrur, Martin Lukasiewycz, Samarjit Chakraborty, Harald Voit, and Anuradha Annaswamy. Challenges in automotive cyber-physical systems design. In 2012 International Conference on Embedded Computer Systems (SAMOS), pages 346–354. IEEE, 2012.

[Gus09] Fredrik Gustafsson. Automotive safety systems. IEEE Signal Processing Magazine, 26(4):32–47, 2009.

[GW09] Minzhe Guo and Ju An Wang. An ontology-based approach to model common vulnerabilities and exposures in information security. In ASEE Southest Section Conference, 2009.

[Har08] John Harrison. Theorem proving for verification. `https://www.cl.cam.ac.uk/~jrh13/slides/cav-09jul08/slides.pdf`, 2008. Intel Corporation.

[HFBPL09] John Hebeler, Matthew Fisher, Ryan Blace, and Andrew Perez-Lopez. Semantic Web Programming. John Wiley & Sons, 2009.

[HM17] Roland E. Haas and Dietmar P. F. Moller. Automotive connectivity, cyber attack scenarios and automotive cyber security. In 2017 IEEE International Conference on Electro Information Technology (EIT), pages 635–639. IEEE, 2017.

[HMS14] Altaf Ahmad Huqqani, Erwin Mann, and Erich Schikuta. Novel concepts for realizing neural networks as services in the sky. Procedia Computer Science, 29:2315–2324, 2014.

[HRM+04] Alan Hevner, Alan R, Salvatore March, Salvatore T, Park, Jinsoo Park, Ram, and Sudha. Design science in information systems research. 28:75, 2004.

[HS10] Martin Homola and Luciano Serafini. Towards formal comparison of ontology linking, mapping and importing. Proc. DL, 10:291–302, 2010.

[HSYC13] Altaf Ahmad Huqqani, Erich Schikuta, Sicen Ye, and Peng Chen. Multicore and gpu parallelization of neural networks for face recognition. Procedia Computer Science, 18:349–358, 2013.

[iec09] IEC 62443-3-1 (TR): Industrial communication networks – network and system security – part 3-1: Security technologies for industrial automation and control systems. https://webstore.iec.ch/publication/7031, 2009.

[IEC15] IEC 62443-3-2. IEC 62443 security for industrial automation and control systems - part 3-2: Security risk assessment and system design. Security Standard Draft, ISA, 2015.

[IEC19] IEC. IEC 62443 - Security for industrial automation and control systems - part 4-2: Technical security requirements for IACS components. Technical report, International Standard, Feb. 2019.

[iII20] Security Levels in ISA-99 / IEC 62443. ISA 99 security levels proposal. https://www.scribd.com/document/129590220/ISA-99-SecurityLevels-Proposal/, 2020.

[Int14] SAE International. Automated driving: levels of driving automation are defined in new sae international standard j3016, 2014.

[isa13] IEC 62443-3-3: Industrial communication networks – network and system security – part 3-3: System security requirements and security levels, 2013.

[ISA18] ISA. The 62443 series of standards: Industrial automation and control systems security. (1-4), 2018.

[ISO09a] ISO 15408, information technology - security techniques - evaluation criteria for IT security Common Criteria - part 1: Introduction and general model, 2009.

[ISO09b] ISO/IEC. ISO/IEC 15408-1:2009-information technology–security techniques–evaluation criteria for IT security–part 1: Introduction and general model. standard, 2009.

[iso10] ISO/IEC 27003: Information technology — security techniques — information security management system implementation guidance, 2010.

[iso11] ISO/IEC 27005: Information technology — security techniques — information security risk management – second edition, 2011.

[ISO12] Information technology — Security techniques — Information security management for inter-sector and inter-organizational communications. International standard, 2012.

[ISO13] Code of practice for information security controls. International standard, International Organization for Standardization - ISO and International Electrotechnical Commission - IEC, Geneva-Switzerland, October 2013.

[ISO14a] Information security management systems: Overview and vocabulary. International standard, International Organization for Standardization - ISO and International Electrotechnical Commission - IEC, Geneva-Switzerland, January 2014.

[ISO14b] Code of practice for protection of personally identifiable information (PII) in public clouds acting as pii processors. International standard, August 2014.

[ISO18] Road vehicles-functional safety-part 2: Management of functional safety, ISO 26262, 2018.

[ISO20a] ISO. Road vehicles — functional safety — part 9: Automotive safety integrity level (ASIL)-oriented and safety-oriented analyses. `https://www.iso.org/obp/ui/#iso:std:iso:26262:-9:ed-1:v1:en`, 2020.

[ISO20b] ISO/SAE DIS 21434. Road vehicles – cybersecurity engineering. `https://www.iso.org/standard/70918.html`, 2020. (Accessed on: February 16, 2020).

[ITPAO13] Japan (IPA) Information-Technology Promotion Agency and Information Security Certification Office. Vulnerability assessment guide for developers. Technical report, 2013.

[JHL15] Hyeon Ae Jang, SH Hong, and MK Lee. A study on situation analysis for asil determination. Journal of Industrial and Intelligent Information, 3(2), 2015.

[JKM+17] Christian Janiesch, Agnes Koschmider, Massimo Mecella, Barbara Weber, Andrea Burattin, Claudio Di Ciccio, Avigdor Gal, Udo Kannengiesser, Felix Mannhardt, Jan Mendling, Andreas Oberweis, Manfred Reichert, Stefanie Rinderle-Ma, WenZhan Song, Jianwen Su, Victoria Torres, Matthias Weidlich, Mathias Weske, and Liang Zhang. The internet-of-things meets business process management: Mutual benefits and challenges. CoRR, abs/1709.03628, 2017.

[joh18] johardi. Cellfie. `https://github.com/protegeproject/cellfie-plugin`, 2018. (Accessed on: November 17, 2020).

[Jos12] Josverwoerd. Digesting big data. `https://blog.bigml.com/2012/11/12/digesting-big-data/`, 2012. (Accessed on: January 17, 2020).

[JS19] Kassu Jilcha Sileyew. Research design and methodology. In Evon Abu-Taieh, Abdelkrim El Mouatasim, and Issam H. Al Hadid, editors, Cyberspace. IntechOpen, 2019.

[KFP+04] Lalana Kagal, Tim Finin, Massimo Paolucci, Navcen Srinivasan, Katia Sycara, and Grit Denker. Authorization and privacy for semantic web services. IEEE Intelligent Systems, 19(4):50–56, 2004.

[KH15] KONE Mohamed Kabir and Kamal Himran. Preserving privacy in semantic web applications. 2015.

[KK14] Sumit Kumar and Suresh Kumar. Semantic web attacks and countermeasures. In Advances in Engineering and Technology Research (ICAETR), 2014 International Conference on, pages 1–5. IEEE, 2014.

[KKA17] Adi Karahasanovic, Pierre Kleberger, and Magnus Almgren. Adapting threat modeling methods for the automotive industry. In Proceedings of the 15th ESCAR Conference, pages 1–10, 2017.

[KKJ18] Reinhard Kloibhofer, Erwin Kristen, and Stefan Jakšić. Safety and security in a smart production environment. In International Conference on Computer Safety, Reliability, and Security, pages 190–201. Springer, 2018.

[KN17] MARTIN KASTEBO and VICTOR NORDH. Model-based security testing in automotive industry. Master's thesis, Department of Computer Science and Engineering - UNIVERSITY OF GOTHENBURG, Gothenburg, Sweden, 2017.

[Koo18] Philip Koopman. Practical experience report: Automotive safety practices vs. accepted principles. In International Conference on Computer Safety, Reliability, and Security, pages 3–11. Springer, 2018.

[KPSD10] Mr Suresh Kumar, Mr Rakesh Kumar Prajapati, Manjeet Singh, and Asok De. Realization of threats and countermeasure in semantic web services. International Journal of Computer Theory and Engineering, 2(6):919, 2010.

[LI16] Aljoscha Lautenbach and Mafijul Islam. The HEAling vulnerabilities to ENhance software security and safety (HEAVENS) project - security models. `https://autosec.se/wp-content/uploads/2018/03/HEAVENS_D2_v2.0.pdf`, 2016.

[LMRM15] Maria Leitner, Zhendong Ma, and Stefanie Rinderle-Ma. A cross-layer security analysis for process-aware information systems. arXiv preprint arXiv:1507.03415, 2015.

[MABK16] Georg Macher, Eric Armengaud, Eugen Brenner, and Christian Kreiner. Threat and risk assessment methodologies in the automotive domain. Procedia computer science, 83:1288–1294, 2016.

[MAG+18] Bruno Mozzaquatro, Carlos Agostinho, Diogo Goncalves, João Martins, and Ricardo Jardim-Goncalves. An ontology-based cybersecurity framework for the internet of things. 2018.

[Mar20] Markus Bartsch, Alexander Bobel, Dr. Brian Niehöfer, Markus Wagner, and Maximilian Wahner. OTP Protection profileof anAutomotive Gateway. `https://unece.org/fileadmin/DAM/trans/doc/2020/wp29/WP29-181-10e.pdf`, 2020.

[McA16] McAfee. Automotive security best practices. Technical report, McAfee, June 2016. 2016.

[MHSP17] Zhendong Ma, Aleksandar Hudic, Abdelkader Shaaban, and Sandor Plosz. Security viewpoint in a reference architecture model for cyber-physical production systems. In Security and Privacy Workshops (EuroS&PW), 2017 IEEE European Symposium on, pages 153–159. IEEE, 2017.

[MIS04] C+ MISRA. Guidelines for the use of the c language in critical systems. MIRA Limited. Warwickshire, UK, 2004.

[MJGA15] Bruno A Mozzaquatro, Ricardo Jardim-Goncalves, and Carlos Agostinho. Towards a reference ontology for security in the internet of things. In Measurements & Networking (M&N), 2015 IEEE International Workshop on, pages 1–6. IEEE, 2015.

[MS16] Zhendong Ma and Christoph Schmittner. Threat modeling for automotive security analysis. Advanced Science and Technology Letters, 139:333–339, 2016.

[MSG+19] Abdelkader Magdy, Christoph Schmittner, Thomas Gruber, A Baith Mohamed, Gerald Quirchmayr, and Erich Schikuta. Cloudwot-a reference model for knowledge-based iot solutions. 2019.

[MSR07] Peter Mell, Karen Scarfone, and Sasha Romanosky. A complete guide to the common vulnerability scoring system version 2.0. `https://www.first.org/cvss/v2/cvss-v2-guide.pdf`, 2007.

[MUT17] ANN STEFFORA MUTSCHLER. Data storage issues grow for cars. `https://semiengineering.com/data-issues-grow-for-cars/`, 2017. (Accessed on: November 11, 2019).

[MV13] Charlie Miller and Chris Valasek. Adventures in automotive networks and control units. Def Con, 21:260–264, 2013.

[MV15] Charlie Miller and Chris Valasek. Remote exploitation of an unaltered passenger vehicle. Technical report, 2015.

[Nat20] United Nations. Proposal for a new un regulation on uniform provisions concerning the approval of vehicles with regards to cybersecurity and cybersecurity management system. `https://undocs.org/ECE/TRANS/WP.29/2020/79`, April 2020. 1958 Agreement: Consideration of proposals for new UN Regulations submitted by the Working Parties subsidiary to the World Forum.

[Net18] Netscribes. The role of iot in the automotive industry. `https://www.netscribes.com/the-present-and-future-role-of-automotive-iot/`, 2018. (Accessed on: February 11, 2020).

[NHT16] NHTSA. Automotive cybersecurity. `https://www.nhtsa.gov/crash-avoidance/automotive-cybersecurity`, 2016. (Accessed on: December 3, 2020).

[NHT17] NHTSA. Automated vehicles for safety. `https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety`, 2017. (Accessed on: February 10, 2020).

[NHT19] NHTSA. Vehicle cybersecurity. https://www.nhtsa.gov/technology-innovation/vehicle-cybersecurity, 2019. (Accessed on: November 27, 2019).

[noa00] Semantic web - XML2000 - slide "architecture". https://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html, 2000. (Accessed on: December 17, 2020).

[noa12] PASTA process for attack simulation and threat analysis (PASTA) risk-centric threat modeling. http://securesoftware.blogspot.com/2012/09/rebooting-software-security.html, 2012. (Accessed on: October 1, 2020).

[noa17a] Common criteria for information technology security evaluation part 1: Introduction and general model. https://www.commoncriteriaportal.org/files/ccfiles/CCPART1V3.1R5.pdf, 2017.

[noa17b] Common criteria for information technology security evaluation part 2: Security functional requirements. https://www.commoncriteriaportal.org/files/ccfiles/CCPART2V3.1R5.pdf, 2017.

[noa17c] Common criteria for information technology security evaluation part 3: Security assurance components. https://www.commoncriteriaportal.org/files/ccfiles/CCPART3V3.1R5.pdf, 2017.

[noa18a] ISO/IEC 27000 – key international standard for information security revised. https://www.iso.org/cms/render/live/en/sites/isoorg/contents/news/2018/03/Ref2266.html, 2018. (Accessed on: October 10, 2020).

[noa18b] New standard specifies capability for control systems - ISA. https://www.isa.org/intech-home/2018/september-october/departments/new-standard-specifies-security-capabilities-for-c, 2018. (Accessed on: October 10, 2020).

[noa20a] BSI - IT-grundschutz "bundesamt für sicherheit in der informationstechnik". https://www.bsi.bund.de/EN/Topics/ITGrundschutz/itgrundschutz_node.html, 2020. (Accessed on: October 10, 2020).

[noa20b] An introduction to qualitative research. https://www.scribbr.com/methodology/qualitative-research/, 2020.

[noa20c] protégé. https://protege.stanford.edu/, 2020. (Accessed on: December 11, 2020).

[Nor16] Alan T Norman. Hacking: Computer Hacking Beginners Guide how to Hack Wireless Network, Basic Security and Penetration Testing, Kali Linux, Your First Hack. CreateSpace Independent Publishing Platform, 2016.

[NVD20a] NIST NVD. NVD - CVSS v2 calculator. https://nvd.nist.gov/vuln-metrics/cvss/v2-calculator, 2020. (Accessed on: October 2, 2020).

[NVD20b] NIST NVD. Vulnerability metrics. https://nvd.nist.gov/vuln-metrics/cvss, 2020. (Accessed on: October 2, 2020).

[O'C19] Martin Joseph O'Connor. Swrlapi. https://github.com/protegeproject/swrlapi, 2019. (Accessed on: December 01, 2020).

[O'C20] Martin O'Connor. MappingMasterDSL. https://github.com/protegeproject/mapping-master/wiki/MappingMasterDSL, 2020. (Accessed on: October 18, 2020).

[OD09] Martin J O'Connor and Amar K Das. SQWRL: A Query Language for OWL. In OWLED, volume 529, 2009.

[Ols16] Mats Olsson. HEAling vulnerabilities to ENhance software security and safety. https://www.vinnova.se/globalassets/mikrosajter/ffi/dokument/slutrapporter-ffi/elektronik-mjukvara-och-kommunikation-rapporter/2012-04625eng.pdf, 2016.

[OML20] OMLAB. Secure tropos. https://austria.omilab.org/psm/content/sectro/info, 2020. (Accessed on: October 9, 2020).

[Ont18] Ontotext. What are ontologies? https://ontotext.com/knowledgehub/fundamentals/what-are-ontologies/, 2018. (Accessed on: January 26, 2020).

[oPF17] Future of Privacy Forum. PERSONAL DATA IN YOUR CAR - national automobile dealers association and the future of privacy forum. https://fpf.org/wp-content/uploads/2017/01/consumerguide.pdf, 2017.

[OSM+08] Martin J O'Connor, Ravi D Shankar, Mark A Musen, Amar K Das, and Csongor Nyulas. The swrlapi: A development environment for working with swrl rules. In OWLED, 2008.

[OVE10] OVERSEE. The application store for cars: Secure download of your favorite apps into your car. https://www.oversee-project.com/fileadmin/oversee/press_releases/OVERSEE-Pressrelease-1-EN.pdf, 2010.

[OWA20] OWASP. Threat modeling - OWASP cheat sheet series. https://cheatsheetseries.owasp.org/cheatsheets/Threat_Modeling_Cheat_Sheet.html, 2020. (Accessed on: October 1, 2020).

[PJP12] K Punithasurya and S Jeba Priya. Analysis of different access control mechanism in cloud. International Journal of Applied Information Systems (IJAIS), Foundation of Computer Science FCS, 4(2), 2012.

[PMRMP18] Florian Pauker, Jürgen Mangler, Stefanie Rinderle-Ma, and Christoph Pollak. Centurio. work-modular secure manufacturing orchestration. 2018.

[Pri20] Princeton University. WordNet: A Lexical Database for English. https://wordnet.princeton.edu, 2020. (Accessed on: November 10, 2020).

[PRZB11] Raluca Ada Popa, Catherine Redfield, Nickolai Zeldovich, and Hari Balakrishnan. Cryptdb: protecting confidentiality with encrypted query processing. In Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles, pages 85–100. ACM, 2011.

[RAP19] RAPID7. Vulnerabilities, exploits, and threats - defining three key terms in cybersecurity. https://www.rapid7.com/fundamentals/vulnerabilities-exploits-threats, 2019. (Accessed on: September 20, 2020).

[Rap20] Rapid7. Vulnerabilities, exploits, and threats, defining three key terms in cybersecurity. https://www.rapid7.com/fundamentals/vulnerabilities-exploits-threats/, 2020. (Accessed on: June 27, 2020).

[RDF] RDF4J. http://rdf4j.org/. (Accessed on: July 18, 2017).

[Ris] Andre Ristaino. Industrial automation cybersecurity conformity assessments. http://www.isasecure.org/en-US/Articles/Industrial-automation-cybersecurity-conformity-ass.

[RPMS17] R Ramesh, M Prabu, S Magibalan, and P Senthilkumar. Hazard identification and risk assessment in automotive industry. International Journal of ChemTech Research, 10(4):352–358, 2017.

[SAE16] SAE. Cybersecurity guidebook for cyber-physical vehicle systems j3061_201601. https://www.sae.org/standards/content/j3061_201601/, 2016. (Accessed on: September 25, 2020).

[SAE18] SAE. Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles j3016_201806. https://www.sae.org/standards/content/j3016_201806, 2018. (Accessed January 19, 2020).

[SBRM17] Torben Stolte, Gerrit Bagschik, Andreas Reschka, and Markus Maurer. Hazard analysis and risk assessment for an automated unmanned protective vehicle. In 2017 IEEE Intelligent Vehicles Symposium (IV), pages 1848–1855. IEEE, 2017.

[SDK19] Florian Sommer, Jürgen Dürrwang, and Reiner Kriesten. Survey and classification of automotive security attacks. 10(4):148, 2019.

[Sec20] Argus Cyber Security. UNECE WP29 Automotive Cyber Security Requirements. https://argus-sec.com/unece-wp29-automotive-cyber-security-requirements/, 2020. (Accessed on: June 11, 2020).

[Ser20] IT/OT Executive Series. /OT Executive Series: What You Need to Know About – Networking. https://www.cogentind.com/it-ot-networking/, 2020. (Accessed on: June 15, 2020).

[SGM18] Christoph Schmittner, Gerhard Griessnig, and Zhendong Ma. Status of the development of iso/sae 21434. In European Conference on Software Process Improvement, pages 504–513. Springer, 2018.

[SGPS14] Christoph Schmittner, Thomas Gruber, Peter Puschner, and Erwin Schoitsch. Security application of failure mode and effect analysis (FMEA). In International Conference on Computer Safety, Reliability, and Security, pages 310–325. Springer, 2014.

[SGS12] Ina Schieferdecker, Juergen Grossmann, and Martin Schneider. Model-based security testing. arXiv preprint arXiv:1202.6118, 2012.

[She18] Nataliya Shevchenko. Threat modeling: 12 available methods. https://insights.sei.cmu.edu/sei_blog/2018/12/threat-modeling-12-available-methods.html, 2018. accessed on: September 27, 2020.

[Shi03] Nobuhisa Shiraishi. The Security of the Semantic Web - Secrecy, Trust and Rationality. https://www.w3.org/People/n-shiraishi/work/Security-of-RDF.html#2.2., 2003. (Accessed on: August 6, 2017).

[Sho08] A. Shostack. Experiences threat modeling at microsoft. volume 413, 2008.

[Sho14] Adam Shostack. Threat modeling: designing for security. Wiley, 2014. OCLC: ocn855043351.

[SKE17] AAmir Shahzad, Young-Gab Kim, and Abulasad Elgamoudi. Secure iot platform for industrial control systems. In Platform Technology and Service (PlatCon), 2017 International Conference on, pages 1–6. IEEE, 2017.

[SKI+19] Nikolaos Serketzis, Vasilis Katos, Christos Ilioudis, Dimitrios Baltatzis, and G Pangalos. Actionable threat intelligence for digital forensics readiness. Information and Computer Security, 27(2):273–291, 2019.

[SKS18a] Abdelkader Magdy Shaaban, Erwin Kristen, and Christoph Schmittner. Application of iec 62443 for iot components. Springer, 2018.

[SKS18b] Abdelkader Magdy Shaaban, Erwin Kristen, and Christoph Schmittner. Application of iec 62443 for iot components. In International Conference on Computer Safety, Reliability, and Security, pages 214–223. Springer, 2018.

[SLAMH18] Christoph Schmittner, Martin Latzenhofer, Shaaban Abdelkader Magdy, and Markus Hofer. A proposal for a comprehensive automotive cybersecurity reference architecture. In The 7th International Conference on Advances in Vehicular Systems, Technologies and Applications, 2018.

[SLSH18] Christoph Schmittner, Martin Latzenhofer, Abdelkader Magdy Shaaban, and Markus Hofer. A proposal for a comprehensive automotive cybersecurity reference architecture. In The Seventh International Conference on Advances in Vehicular Systems, Technologies and Applications, pages 30–36, 2018.

[SM13] Erich Schikuta and Erwin Mann. N2sky—neural networks as services in the clouds. In Neural Networks (IJCNN), The 2013 International Joint Conference on, pages 1–8. IEEE, 2013.

[SM14] Christoph Schmittner and Zhendong Ma. Towards a framework for alignment between automotive safety and security standards. In International Conference on Computer Safety, Reliability, and Security, pages 133–143. Springer, 2014.

[SMH+17] Erich Schikuta, Abdelkader Magdy, Irfan Ul Haq, A Baith Mohamed, Benedikt Pittl, and Werner Mach. Searching the sky for neural networks. In International Work-Conference on Artificial Neural Networks, pages 167–178. Springer, 2017.

[Smi16] Craig Smith. The car hacker's handbook: a guide for the penetration tester. no starch press, 2016.

[SMM16] Erich Schikuta, Abdelkader Magdy, and A. Baith Mohamed. A framework for ontology based management of neural network as a service. In 23rd International Conference on Neural Information Processing (ICONIP 2016), October 2016.

[SMR+16] Christoph Schmittner, Zhendong Ma, Carolina Reyes, Oliver Dillinger, and Peter Puschner. Using SAE j3061 for automotive security requirement engineering. In Amund Skavhaug, Jérémie Guiochet, Erwin Schoitsch, and Friedemann Bitsch, editors, Computer Safety, Reliability, and Security, volume 9923, pages 157–170. Springer International Publishing, 2016. Series Title: Lecture Notes in Computer Science.

[SMS14] Christoph Schmittner, Zhendong Ma, and Paul Smith. FMVEA for safety and security analysis of intelligent and cooperative vehicles. In International Conference on Computer Safety, Reliability, and Security, pages 282–288. Springer, 2014.

[SNP+16] Adam Sotona, Stefan Negru, MSDIT Prague, et al. How to feed Apache HBase with petabytes of RDF data: An extremely scalable RDF store based on Eclipse RDF4J framework and Apache HBase database. In Proceedings of the ISWC, 2016.

[Som07] I. Sommerville. Software Engineering. International computer science series Software engineering. Addison-Wesley, 2007.

[Spa17] Sparxsystems. Enterprise Architect. `http://www.sparxsystems.com/products/ea/`, 2017. (Accessed on: July 14, 2017).

[SRAS19] Esmaeili Seyed Reza and Esterabadi Afshin Soltani. Attack analysis methodologies. `https://odr.chalmers.se/bitstream/20.500.12380/300610/1/CSE%2019-103%20ODR%20Esmaeili%20Esterabadi.pdf`, 2019. Master dissertation.

[SRI16] S SRIDHAR. A literature survey on automobile safety, practice of design and regulation. 2016.

[SS04] Frank Swiderski and Window Snyder. Threat Modeling. Microsoft Press, 2004.

[SS05] A Stamos and S Stender. Attacking web services: The next generation of vulnerable enterprise apps. BlackHat2005, pages 1–20, 2005.

[SS13] Steffen Staab and Rudi Studer. Handbook on ontologies. Springer Science & Business Media, 2013.

[SSG+19] Abdelkader Magdy Shaaban, Christoph Schmittner, Thomas Gruber, A. Baith Mohamed, Gerald Quirchmayr, and Erich Schikuta. Ontology-based model for automotive security verification and validation. In Proceedings of the 21st International Conference on Information Integration and Web-Based Applications & Services, iiWAS2019, page 73–82, New York, NY, USA, 2019. Association for Computing Machinery.

[SSMCW15] Amina Souag, Camille Salinesi, Raúl Mazo, and Isabelle Comyn-Wattiau. A security ontology for security requirements elicitation. In International symposium on engineering secure software and systems, pages 157–177. Springer, 2015.

[SSMG16] Erwin Schoitsch, Christoph Schmittner, Zhendong Ma, and Thomas Gruber. The need for safety and cyber-security co-engineering and standardization for highly automated automotive vehicles. In Advanced Microsystems for Automotive Applications 2015, pages 251–261. Springer, 2016.

[SSQ+19] Abdelkader Magdy Shaaban, Christoph Schmittner, Gerald Quirchmayr, A Baith Mohamed, Thomas Gruber, and Erich Schikuta. Toward the ontology-based security verification and validation model for the vehicular domain. In International Conference on Neural Information Processing, pages 521–529. Springer, 2019.

[SSWM13] Amina Souag, Camille Salinesi, Isabelle Wattiau, and Haris Mouratidis. Using security and domain ontologies for security requirements analysis. In 2013 IEEE 37th Annual Computer Software and Applications Conference Workshops, pages 101–107. IEEE, 2013.

[STH+19] Christoph Schmittner, Peter Tummeltshammer, David Hofbauer, Abdelkader Magdy Shaaban, Michael Meidlinger, Markus Tauber, Arndt Bonitz, Reinhard Hametner, and Manuela Brandstetter. Threat modeling in the railway domain. In International Conference on Reliability, Safety, and Security of Railway Systems, pages 261–271. Springer, 2019.

[Syn19] Synopsys. The 6 levels of vehicle autonomy explained. `https://www.synopsys.com/automotive/autonomous-driving-levels.html`, 2019. (Accessed on: June 27, 2020).

[Sys19] SWINDON Silicon Systems. Determining functional safety levels for automotive applications. https://www.swindonsilicon.com/determining-functional-safety-levels-for-automotive-applications/, 2019. accessed January 25, 2020.

[Sys20] Enterprise Architect Sparx Systems. Model driven generation (mdg) technologies. `http://www.sparxsystems.com/resources/mdg_tech/`, 2020. (Accessed on: June 12, 2020).

[TC20] The Task Force on and Cyber Security and Over-the-Air issues. Proposal for amendments to ECE/TRANS/WP.29/GRVA/2020/3 - draft new UN regulation on uniform provisions concerning the approval of vehicles with regard to cyber security and of their cybersecurity management systems, 2020.

[Tch02] Lubka Tchankova. Risk identification–basic stage in risk management. Environmental management and health, 2002.

[Tes20] Tesla. Autopilot. `https://www.tesla.com/autopilot`, 2020. (Accessed on: October 10, 2020).

[Thu15] Bhavani Thuraisingham. Security standards for the semantic web. Computer Standards & Interfaces, 02, 2015.

[Til20] Fischer Till. Overview of ISO/SAE 21434 – YSEC. https://www.security-analyst.org/iso-sae-21434/, 2020. (Accessed on: October 10, 2020).

[TSL+06] Marie-Noölle Terrasse, Marinette Savonnet, Eric Leclercq, Thierry Grison, and George Becker. Do we need metamodels and ontologies for engineering platforms? In Proceedings of the 2006 international workshop on Global integrated model management, pages 21–28, 2006.

[TW07] Stephen Thomas and Laurie Williams. Using automated fix generation to secure sql statements. In Proceedings of the Third International Workshop on Software Engineering for Secure Systems, page 9. IEEE Computer Society, 2007.

[UNE17a] UNECE. Recommendation of the Task Force on Cyber Security and Over-the-air issues of UNECE WP.29 IWG ITS/AD on Cyber Security. https://wiki.unece.org/pages/viewpage.action?pageId=46792870, October 2017. (Accessed on: January 17, 2020).

[UNE17b] United Nations Economic Commission for Europe UNECE. CSOTA ad hoc "threats 2". https://wiki.unece.org/download/attachments/45383725/TFCS-ahT2-06%20%28Chair%29%20Table%20on%20CS%20threats%20-%20changes%20agreed%20by%20ahT2%20-%20non-cleaned%20up.xlsx?api=v2, 2017. (Accessed on: June 06, 2020).

[UNE18] UNECE WP.29. Status report of TF-CS/OTA. https://slideplayer.com/slide/16280109/, 2018. (Accessed on: June 12, 2020).

[UNE19] UNECE. Un transport agreements and conventions. http://www.unece.org/trans/maps/un-transport-agreements-and-conventions-18.html, 2019. (Accessed on: June 12, 2020).

[UNE20a] UNECE. Proposal for amendments to ECE/TRANS/WP.29/GRVA/2020/3. Technical report, United Nations Economic Commission for Europe (UNECE), February 2020. The provisions proposed in para. 5.3.1. through 5.3.4. and 7.4. are in square brackets.

[UNE20b] UNECE. Working Party on Automated/Autonomous and Connected Vehicles (GRVA). https://www.unece.org/trans/main/wp29/meeting_docs_grva.html, 2020. (Accessed on: June 11, 2020).

[UNE20c] UNECE. World Forum for Harmonization of Vehicle Regulation (WP.29). https://www.unece.org/trans/main/wp29/meeting_docs_wp29.html, 2020. Accessed on: June 10, 2020).

[UNE20d] UNECE. Wp.29 - introduction. http://www.unece.org/trans/main/wp29/introduction.html, 2020. (Accessed on: June 11, 2020).

[UPP19] UPPAAL. Uppaal help. https://www.it.uu.se/research/group/darts/uppaal/help.php?file=Introduction.shtml, 2019. (Accessed on: July 27, 2020).

[vRB16] Timo van Roermund and Andy Birnie. A multi-layer vehicle security framework. Whitepaper, May, pages 11–25, 2016.

[W3C] World Wide Web Consortium (W3C). `https://www.w3.org`. Accessed: 2017-07-11.

[W3C19] W3C. Vocabularies. `https://www.w3.org/standards/semanticweb/ontology`, 2019. (Accessed on: February 12, 2020).

[WDRTCfA11] & the European Organisation for Civil Aviation Equipment. Washington DC: Radio Technical Commission for Aeronautics, Inc. Do -178c: Software considerations in airborne systems and equipment certification. Technical report, 2011.

[Wil94] Theodore J Williams. The purdue enterprise reference architecture. Computers in industry, 24(2-3):141–158, 1994.

[Wil18] Imano Williams. An ontology based collaborative recommender system for security requirements elicitation. In 2018 IEEE 26th International Requirements Engineering Conference (RE), pages 448–453. IEEE, 2018.

[WP08] Marko Wolf and Christof Paar. Security requirements engineering in the automotive domain: On specification procedures and implementational aspects. pages 485–498, 2008.

[WRWW05] by IgorMozetic W3C RIF-WG Wiki. Horn logic. `https://www.w3.org/2005/rules/wg/wiki/Horn_Logic`, 2005. (Accessed on: August 21, 2020).

[YCZZ11] Xiaohu Yang, Yixi Chen, Wenyu Zhang, and Shuai Zhang. Exploring injection prevention technologies for security-aware distributed collaborative manufacturing on the semantic web. The International Journal of Advanced Manufacturing Technology, 54(9):1167–1177, 2011.