electronics

MDPI

Article

Security Risk Analysis Approach for Safety-Critical Systems of Connected Vehicles

Feng Luo, Shuo Hou *, Xuan Zhang, Zhenyu Yang and Wenwen Pan

School of Automotive Studies, Tongji University, Shanghai 201804, China; luo_feng@tongji.edu.cn (F.L.); zhangxuan@tongji.edu.cn (X.Z.); 1811023@tongji.edu.cn (Z.Y.); 1831626@tongji.edu.cn (W.P.)

* Correspondence: houshuo_2015@tongji.edu.cn

Received: 6 July 2020; Accepted: 31 July 2020; Published: 2 August 2020

Abstract: Modern vehicles are no longer merely mechanical systems but are monitored and controlled by various electronic systems. Safety-critical systems of connected vehicles become vulnerable to cyberattacks because of increasing interconnection. At present, the security risk analysis of connected vehicles is mainly based on qualitative methods, while these methods are usually subjective and lack consideration for functional safety. In order to solve this problem, we propose in this paper a security risk analysis framework for connected vehicles based on formal methods. Firstly, we introduce the electronic and electrical architecture of the connected vehicle and analyze the attack surfaces of the in-vehicle safety-critical systems from three levels of sensors, in-vehicle networks, and controllers. Secondly, we propose a method to model the target of evaluation (i.e., in-vehicle safety-critical system) as a Markov decision process and use probabilistic computation tree logic to formally describe its security properties. Then, a probabilistic model checker PRISM is used to analyze the security risk of target systems quantitatively according to security properties. Finally, we apply the proposed approach to analyze and compare the security risks of the collision warning system under a distributed and centralized electrical and electronic architecture. In addition, from a practical point of view, we propose a Markov model generation method based on a SysML activity diagram, which can simplify our modeling process. The evaluation results show that we can have a quantitative understanding of the security risks at the system level in the early stage of system design.

Keywords: connected vehicle; safety-critical system; security risk analysis; Markov decision process; probabilistic model checking

1. Introduction

Modern vehicles have changed from a traditional mechanical system to an electronic control system, which runs a large amount of software and hardware [1]. In some high-end vehicles, more than 100 electronic control units (ECUs) are assembled to implement complex safety and comfort functions [2]. It can be predicted that, with the development of autonomous vehicles, the number of electronic components running on vehicles will continue to grow. Nevertheless, the increasing usage of information and communication technology (ICT) introduces new safety and security issues. Security threats from inside and outside the vehicle cannot only damage to the privacy of the system but also the safety of life, such as when the dynamic control system of the vehicle is under the control of an attacker.

The stable operation of automotive safety-critical systems is the basis of ensuring the safety of drivers and passengers. With the increasing connectivity of vehicles, some non-safety-critical systems have become safety-critical systems inside the connected vehicles. For example, when the driver's seat suddenly slides backward, if the driver wants to brake in case of an emergency, the driver is likely to be unable to complete the needed action. In the past, automotive engineers devoted much attention to the research of safety-critical systems to prevent these systems from failure, but they lacked experience in ensuring the security of these systems. The safety-critical system of the vehicle is composed of various electronic components, such as controllers, sensors, and actuators. These components are connected together by various internal networks, such as controller area networks (CANs), local interconnect networks (LINs), and FlexRay. These in-vehicle networks were originally designed for a closed network environment; as a result, the increasing attack surfaces make in-vehicle networks vulnerable to malicious attacks. Experimental security research in [3–5] has shown that malicious attacks against safety-critical systems are possible. Therefore, a complete set of security defense methods needs to be studied urgently.

The functions implemented by the software on the vehicles are increasing at a rate of about 30% every year, which makes the manageability of the vehicle's functions a great challenge (e.g., real-time, safety, and security) [6]. With the development of advanced driving assistance systems (ADASs), automatic driving technology, and V2X technology, the security threat scenarios faced by vehicles have become more complex and have a serious impact. Therefore, a systematic method is urgently needed to evaluate the cybersecurity threats of safety-critical systems inside connected vehicles. Although J3061 [7] has recommended many guiding methods for security analysis of vehicles, these methods are subjective and lack quantitative analysis results. In the early stage of automotive development, a system-level quantitative security threat analysis and risk assessment method are still needed.

1.1. Related Works

Related research has shown that in-vehicle networks can be intruded through various external networks (e.g., Cellular, Wi-Fi, Bluetooth, and FM radio) [3,5,8,9]. As described in [10], vehicle-embedded systems can be divided into safety-critical systems and non-safety-critical systems. It is worth noting that, with the development of vehicle connectivity, the cybersecurity of electronic components in the vehicle will affect the functional safety of the vehicle. An attacker can access the infotainment system by using a smartphone and then access the CAN bus to control the in-vehicle dynamic control systems [11]. A malicious attacker can initiate a denial of service (DoS) attack by using the CAN bus protocol vulnerability, resulting in CAN network paralysis [12]. More and more attention has been paid to the privacy security, financial security, and even functional safety of in-vehicle information systems. In order to solve these problems, many researchers have carried out research on security mechanisms, such as a vehicle security protocol, firewall, and intrusion detection systems (IDS) [13–15]. These security mechanisms are inspired by IT systems and are tailored and optimized for use in vehicles. However, a significant problem is which key component of the vehicle should be deployed with the security mechanism to achieve strong security at the lowest cost. A feasible method to solving this problem is to analyze the threats of the target system in the early stage of security design, determine the security risk of the system, and then implement different security strategies according to the security risk level.

There are many standards and frameworks for threat analysis and risk assessment, but they were designed for IT systems and cannot meet the requirements of vehicles. SAE J3061 was the first special guide to solve the security issues of cyber-physical vehicle systems [7]. It refers to the hazard analysis process of ISO 26262 to put forward the security analysis process of automotive electronic systems. However, it only standardizes the process and does not really put forward a method that can be implemented in combination with the ISO 26262 standard. The EVITA method, recommended in SAE J3061, is the first specifically designed method for vehicle threat analysis [16]. In the conceptual design stage, the EVITA method evaluates four aspects of the automotive electronic and electrical system: functionality, safety, privacy, and operational severity, and finally determines its security level. This method has numerous evaluation indexes and increases the difficulty of evaluation. Its core idea is to use an attack tree to model the threat scene, which is easy to use but rather subjective. The HEAVENS method, another threat analysis method mentioned in SAE J3061, introduces the Microsoft STRIDE model into the threat analysis of the automotive electronic system [17]. For each threat, the HEAVENS method determines its risk level according to the threat level (TL) and impact level (IL). The HEAVENS method simplifies the indicators of threat analysis and is easier to implement. In addition to the threat analysis methods mentioned in SAE J3061, many other methods (such as OCTAVE, ATA, and FMVEA) that may be applicable to the automotive environment are summarized in [6,18]. Despite all this, these methods are qualitative analyses and have not been effectively verified in the automotive context by considering functional safety. Wang et al. [19] proposed a security modeling method for a cooperating vehicle platoon in an open-access environment, which takes safety-related cyberattacks into account but lacks detailed in-vehicle analysis. Hamad et al. [20] presented an attack-tree-based hybrid threat modeling method for vehicular systems by combining existing approaches but without quantitative analysis. Carreras Guzman et al. [21] proposed a multi-layered representation of cyber-physical systems (CPSs), which provides a possible method to conduct combined safety and security risk analysis of an autonomous vehicle.

In order to solve the problem of quantitative evaluation, Kong et al. [22] proposed an attack-tree-based security risk assessment framework for smart cars. In the framework, the attack tree is used to calculate the probability of attack events by applying a resistance formula from circuit theory. The limitation of this study was that it was based on the assumption that the technical capabilities and costs of attackers are the same. Because automotive functional safety and cybersecurity are closely related, since the functional safety has been a well-established process, it is meaningful to combine the cybersecurity and functional safety standards. Macher et al. [23] put forward a risk analysis method, called SAHARA, that combines the HARA of functional safety and Microsoft's STRIDE security threat model. SAHARA defined security level (SecL) for the safety-critical system of vehicles based on the ASIL defined in ISO 26262. It was shown that the SAHARA method could identify more hazardous situations than the conventional HARA method. However, the qualitative analysis method used by SAHARA is insufficiently fine-grained in the description of the risk level. Summarizing the current research status, some problems still need to be further addressed in the analysis of automotive security threats, including modeling the analysis process, improving the degree of quantification, and integrating automotive functional safety. Mundhenk et al. [2] presented a security analysis process of automotive architectures based on the continuous-time Markov chain (CTMC) model, which enables quantifying the security risk of various properties. However, the CTMC model cannot describe the non-deterministic behavior of the attacker. Mohsin et al. [24] proposec the IoTRiskAnalyzer framework to quantitatively analyze the security risk of the Internet of Things (IoT), in which the Markov decision process (MDP) is used to model the non-deterministic behavior of the attacker. Inspired by these work, we propose a security risk analysis approach based on the MDP model for safety-critical systems of connected vehicles.

1.2. Paper Contributions

The key contribution of this paper is to present a formal security threat analysis and risk assessment framework for in-vehicle safety-critical systems. The benefits of this paper's work are multifold:

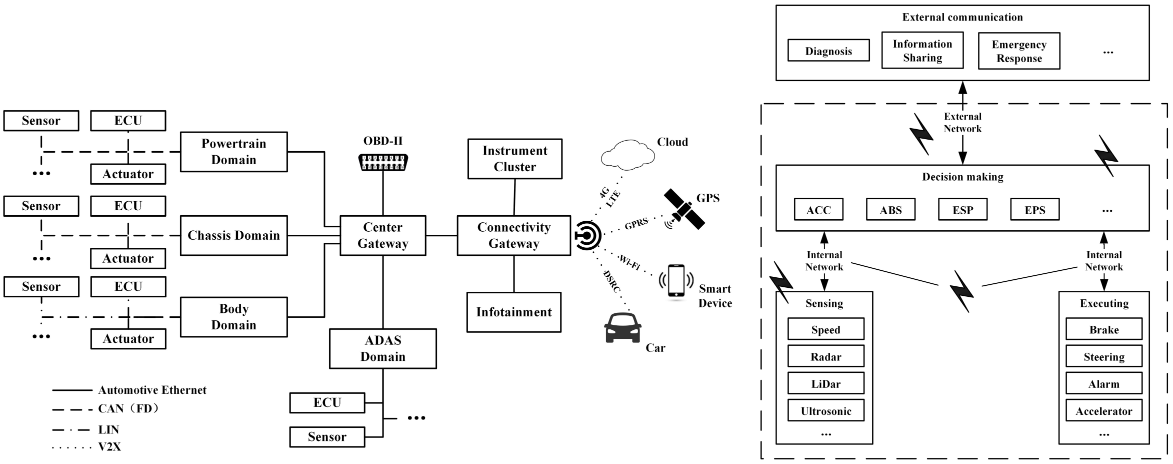
- Summarizing the attack surfaces of safety-critical systems inside connected vehicles and dividing these attack surfaces into three categories: sensing attacks, in-vehicle network attacks, and controller attacks. This classification method makes it easy to understand the security threats faced by the safety-critical systems of connected vehicles.
- A formal security threat analysis and risk assessment framework based on the Markov decision process model is proposed, which can be used for quantitative security analysis of safety-critical systems at the system-level. This model-based security threat analysis method reduces the subjectivity of the assessment process.
- The proposed approach is applied to the security threat analysis of the specific in-vehicle safety-critical systems, and the evaluation results show the different security risks between traditional distributed and modern centralized electrical and electronic architecture (EEA).

1.3. Paper Organization

The rest of this paper is organized as follows. Section 2 summarizes the attack surfaces of the safety-critical systems of connected vehicles. Section 3 introduces the concepts of Markov decision process and probabilistic model checking. Section 4 presents the proposed security threat analysis and risk assessment framework. Section 5 presents the illustration. Finally, Section 6 concludes this paper.

2. Attack Surfaces Analysis

The automobile is evolving from a traditional mechanical system to an autonomous driving system, alongside the innovation of automotive electrical and electronic architecture. More ECUs, sensors, and actuators are being added to the vehicle system for safety functions, automatic driving, and so forth. A very promising trend of automotive EEA innovation is that the automotive Ethernet serves as the backbone in vehicles, including a domain-based architecture, to meet the needs of mass data transmission of various components inside the vehicle, as well as external connected devices and equipment. Figure 1a illustrates the recently proposed centralized automotive EEA, in which the automotive Ethernet serves as the backbone. In this EEA, the domain controllers divide the in-vehicle networks into different subsystems, including powertrain, chassis, body, and ADAS. Each subsystem includes many electronic components and uses different bus systems for internal communication according to different functions. The domain controllers are connected to the in-vehicle gateway via the Ethernet backbone and further connected to the external network through the connectivity gateway.



(**a**) Automotive Ethernet serves as backbone. (**b**) Attack surfaces of safety-critical systems.

Figure 1. Automotive electrical and electronic architecture (EEA) and attack surfaces.

The connected vehicle is a complex system, which integrates environment perception, decision-making, and executing, and depends on continuous information exchange among sensors, actuators, ECUs, and external equipment. In general, the ECU receives the sensor signal and then makes intelligent decisions and sends the control signal to the actuator. Furthermore, the connectivity of vehicles increases the intelligence level of the vehicle. On the one hand, V2X technology helps vehicles make better decisions and improve safety by getting traffic information from external infrastructures. On the other hand, vehicles will provide remote access functions to realize remote start, diagnosis, emergency response, and so forth. However, the open vehicle network architecture increases the security risks of the in-vehicle network system. The safety operation of vehicles depends on the cybersecurity of the in-vehicle network system to a great extent. Many studies in the literature presented vulnerabilities and threat analyses of in-vehicle networks, ECUs, and V2X communication. Pettit et al. [4] analyzed the attack surfaces in autonomous and automated vehicles. For safety-critical systems inside connected vehicles, as shown in Figure 1b, we have summarized corresponding cyberattacks and divided them into three categories.

Sensing attacks: A modern vehicle is equipped with various acoustic, optical, and electromagnetic sensors for perceiving the surrounding environment to realize the auxiliary driving function. It has been shown that remote blinding and jamming attacks against autonomous vehicles' sensors (e.g., GPS, LiDAR, radar, and cameras) are possible. For collaborative vehicles, data acquired through V2X from outside equipment, such as roadside units (RSU) and other vehicles, can be regarded as a special kind of "sensing". Information from the special "sensor" and in-vehicle sensors is fused together to perceive as much environmental information as possible. Attacks against the environment perception process will cause the controller to get the wrong information and then make a wrong decisions.

In-vehicle network attacks: The on-board ECUs communicate with each other through various in-vehicle networks, which were originally designed for a closed network environment, and thus no security mechanism was considered. Both engine and transmission systems are controlled by ECUs by using the CAN bus. Therefore, the in-vehicle CAN bus is a high-priority target for attackers, which can lead to serious malfunctions and accidents. It is worth noting that automotive Ethernet was introduced to meet high bandwidth requirements while introducing its inherent vulnerability. Because the gateway providing the connected function, attackers can use the external network connection interface to attack the backbone to jam or tamper with sensors and control signals.

Controller attacks: As mentioned above, a modern vehicle's function is usually controlled by a number of ECUs. In order to facilitate software upgrades, these ECUs' firmware can usually be refreshed through the unified diagnostic services (UDS) protocol. Because of the weak security algorithm during firmware updates, attackers can easily inject malicious firmware into ECUs. In recent years, more powerful ECUs were developed for the ADAS system and connected gateway, usually along with a Linux-based operating system. A new emerging risk is that a back-door program may be implanted into these ECUs. Then, attackers can gain unauthorized access priority, which will damage the system function or cause the remote server to be unable to receive the message of the vehicle.

3. Preliminaries

In this section, we introduce the Markov decision process (MDP) for formalizing probabilistic and non-deterministic behavior and probabilistic model checking technology for security property verification based on a probabilistic computation tree (PCTL).

3.1. Markov Decision Process

Discrete-time markov chains (DTMC) and continuous time markov chains (CTMC) are often used to model systems with probability properties, and the Markov decision process (MDP) extends their ability to model non-deterministic systems. Because of the non-deterministic attacker's behavior and security state transition of the in-vehicle network system, we model our system as an MDP model.

Definition 1 (Markov Decision Process)**.** *A Markov decision process is a 5-tuple* $M = (S, s_0, A, L, Steps)$*, where:*

- S *is a finite set of states;*
- $s_0 \in S$ *is the initial state;*
- A *is a finite set of actions;*
- $L : S \to 2^{AP}$ *is function labeling states with atomic propositions (AP); and*
- $Steps : S \times A \times S \to [0,1]$ *is probabilistic transition function. For each* $s \in S$ *and* $a \in A$*, fulfill* $\sum_{s' \in S} Steps(s, a, s') = 1$*.*

3.2. Probabilistic Model Checking

Model checking is a very important automatic verification technology, the basic idea of which is to use a state transition system M to represent the behavior of the system, and use a temporal logic formula (f) to describe the nature of the system. For instance, the question "does the system have the desired properties" is transformed into the mathematical problem "is the state transfer system M

a model of formula (f)”, which is expressed as $M \models f$ by the formula. Commonly used logic formulas for describing the system properties include LTL, CTL, PCTL, etc. For our purposes, we use PCTL to describe the security properties of target systems with probability properties. Formally, the syntax structure of PCTL can be represented by the BNF paradigm as follows:

$$\phi ::= true|a|\phi \wedge \phi|\phi \vee \phi|\neg\phi|\phi \rightarrow \phi|P_{\sim p}[\psi]$$

$$\psi ::= X\phi|F\phi|G\phi|\phi U^{\leq k}\phi|\phi U\phi$$

where ϕ stands for state formulae, ψ stands for path formulae, $true$ is tautology, $a \in AP$ is an atomic proposition, $\wedge$ is a conjunction connective, $\vee$ is a disjunction connective, $\neg$ is a negation connective, $\sim \in \{<, >, \leq, \geq\}, k \in \mathbb{N}$, $P_{\sim p}[\psi]$ represents the set of states satisfying the path formula ψ under the probabilistic constraint of p ($p \in [0, 1]$), X is “next state”, F is “eventually”, G is “globally”, U is “until”, and $U^{\leq k}$ is “bounded until”.

There are a number of tools available for automatic probabilistic model checking, such as CADP, MRMC, PAT, and PRISM. The model checker takes the formally described system by modeling language and the system properties specification as input; then the given system’s properties can be verified and analyzed in a quantitative way. PRISM supports many probabilistic models, such as CTMC, DTMC, PTA, and MDP, and provides a GUI interface; thus we chose the PRISM tool to verify our system modeled by MDP.

4. Proposed Approach

The proposed security risk assessment framework is depicted in Figure 2. The analysis process of the framework we proposed includes three steps. Firstly, the MDP model of the system is established according to the system model (i.e., topology and functions are described in the system specification documentation) and the identified security threats. Secondly, the security properties described in the system specification with natural language are formalized as PCTL expressions. Finally, the well-coded MDP model and PCTL expressions are imported into the PRISM probability model checker to verify the probability of each security property.

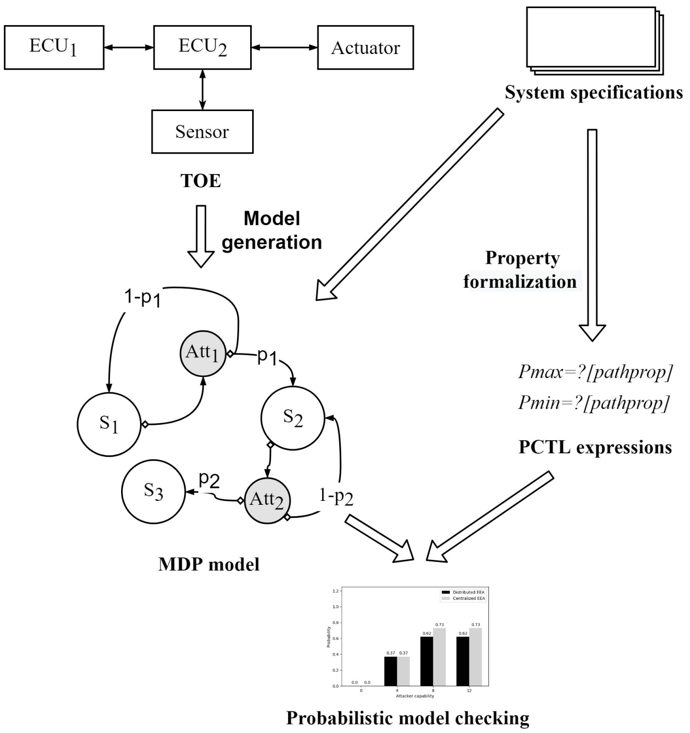


Figure 2. Security risk assessment framework for safety-critical systems of connected vehicles.

4.1. System Model

As mentioned above, the automotive safety-critical system usually consists of various sensors, ECUs, actuators, and communication networks responsible for information exchange between these components. An automotive safety-critical system can be formalized as a quadruple $< N, F, L, M >$, where:

- $N = \{E, S, A, C, D\}$ is the set of nodes comprising a safety-critical system where E, S, A, C, and D represent the set of ECUs, sensors, actuators, cloud server, and external devices, respectively.
- F is the set of functions provided by each component. For example, a sensor can observe the vehicle speed, tire pressure and collision, the in-vehicle gateway can forward the control signal and sensor signal, and actuators can perform the brake, acceleration, and deceleration actions according to the control signals.
- L is the set of automotive onboard links. Differently from the computer network, the components on the vehicle are linked together through a variety of networks, such as CAN, LIN, Ethernet, Wi-Fi, Bluetooth, DSRC, and Cellular.
- $M : N \times N \to L$ is a mapping denoting the communication link between two components.

It is worth noting that we made an abstraction of the functions of the components that comprise the automotive safety-critical system. Firstly, the sensor's perception of the environment may be a continuous signal. However, when the controller makes a decision, it usually makes a judgment on a threshold value. Therefore, we think of sensors as components that only detect and output boolean values. For example, if the ECU needs to make a decision according to a threshold of 10 m from a distance sensor, then we express that the sensor's output is 1 when the distance is less than 10 m, otherwise 0. According to the sensor signal, the controller will make decisions and output the control signal based on the predefined strategies. Secondly, the control signal is also a boolean value, indicating whether the actuator needs to perform corresponding actions. Finally, the actuator completes the action according to the received control signal. We think this abstraction is reasonable because, from the perspective of risk analysis, we do not need to care about the details of how the data is processed, but only whether the data is likely to be attacked. This abstraction can reduce the complexity of risk analysis to a large extent, which will be illustrated in the following case study.

4.2. Threat Model

Vulnerabilities and threats identification: For a given in-vehicle network system, vulnerabilities identification includes determining the existence and exploitability of flaws or weaknesses, and threats refer to an attacker putting the system at risk by exploiting vulnerabilities. In many risk assessment frameworks, vulnerability identification is the first step of risk analysis. However, it is impossible to identify all potential vulnerabilities in the conceptual design stage, because some of them are generated in the conceptual design stage and others are generated in the process of software and hardware development. Therefore, more potential vulnerabilities can only be found through penetration testing and other means. In order to solve this problem, we suggest that threats rather than vulnerabilities should be given priority in the process of risk analysis in the concept stage. In our approach, we regard each possible threat as an atomic attack, and the security risk of the whole system is the result of multiple atomic attacks. We build the scoring method of potential attacks based on the component of potential attacks and the possible classification of EVITA. Usually, vague words such as "Basic", "Enhanced-Basic", "Moderate", "High", and "Beyond High" are used to describe the possibility of an attack. In fact, what we give is an estimation without quantification, that is, different people may not mean the same when they say "Moderate". In order to conduct quantitative risk analysis, we propose to assign a probability value $p \in [0, 1]$ to each attack according to Kent's words of estimative probability [25], as shown in the Table 1.

Assets evaluation: For the safety-critical system of the vehicle, the attacks at the network layer will do harm to the functional safety of the target system. In the functional safety standard ISO 26262 of the automotive electronics industry, it is defined that functional safety means avoiding unreasonable risks caused by electrical and electronic system failures. In ISO 26262, the functional safety level of the system is defined by automotive safety integration level (ASIL), which includes four levels: A, B, C, and D. Among them, A is the lowest safety level and D is the highest. We can identify the various assets contained in the target system from the system model. The higher the ASIL level of an asset, the more important we think it is. Therefore, if the asset's ASIL level is higher, the attacker will cause more serious damage to the asset, or in other words, the more reward the attacker will gain from the attack. It is worth noting that we assume that the target system always meets the functional safety requirements before performing a security risk analysis. In order to evaluate the importance of assets quantitatively, we defined the value of assets based on ISO 26262 ASIL as shown in Table 2.

Attacker model As mentioned above, before threat analysis, we identify various attack points in the system, which are called atomic attacks. The security risk of the target system is determined by the attacker's usage of these atomic attacks. The behavior of an attacker can be represented by Figure 3. For an atomic attack, the probability of success is p, and the attacker will get some reward, which is the value of the corresponding asset denoted by Val_i. The failure probability of an atomic attack is $1 - p$, and the attacker does not get any reward in this case. It is worth noting that the attacker's choice of an atomic attack is non-deterministic. We assume that an attacker always has limited capability denoted by the variable Cap. Therefore, in the process of an attack, the reward gained by the attacker is not greater than his/her capability. For each atomic attack, the reward obtained by the attacker is modeled as the variable $Reward(0 \leq Reward \leq Cap)$. The probability value and reward of a single atomic attack can be quantified according to Tables 1 and 2. The initial value of the reward is set to zero and then increases gradually with the success of the atomic attack. When the attacker's reward reaches his/her personal capability, the attack process will stop.

Table 1. Attack probability value scale.

Attack Potential	Attack Probability	Kent's Estimative Terms	Probability Values
None	-	Certain	100%
Basic	5	Almost Certain	93% (±6%)
Enhanced-Basic	4	Probable	75% (±12%)
Moderate	3	Chances About Even	50% (±10%)
High	2	Probably Not	30% (±10%)
Beyond High	1	Almost Certainly Not	7% (±5%)
Infinite	-	Impossible	0

Table 2. Assets value based on ISO 26262 ASIL.

ISO 26262 ASIL	Assets Value
A	1
B	2
C	3
D	4

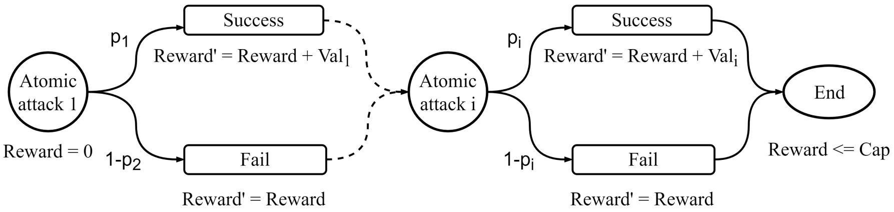


Figure 3. Attacker model.

4.3. MDP Model for Security Risk Analysis

The MDP model for risk analysis is a combination of the system model and threat model, as shown in Figure 4. We built the MDP model in a modular way, including a sensor module, a controller module, an actuator module, and an attacker module. The state transitions (i.e., signal changes) of the system are described in each module, which is driven by the atomic attacks Att_i identified in each module. In particular, the attacker module describes the non-deterministic behavior when the attacker chooses different atomic attacks. As mentioned earlier, the sensor module is a collection of signals used to sense changes of the environment, including real sensors and remote messages received through V2X communication. Different modules are connected by in-vehicle networks, which describe the direction of message transmission. For example, a change of environment will change sensor output data, and then the sensor data will be transmitted to the controller through the network. Due to the participation of the attacker, the signal received by the controller may not reflect the real change of environment and therefore make a wrong decision.

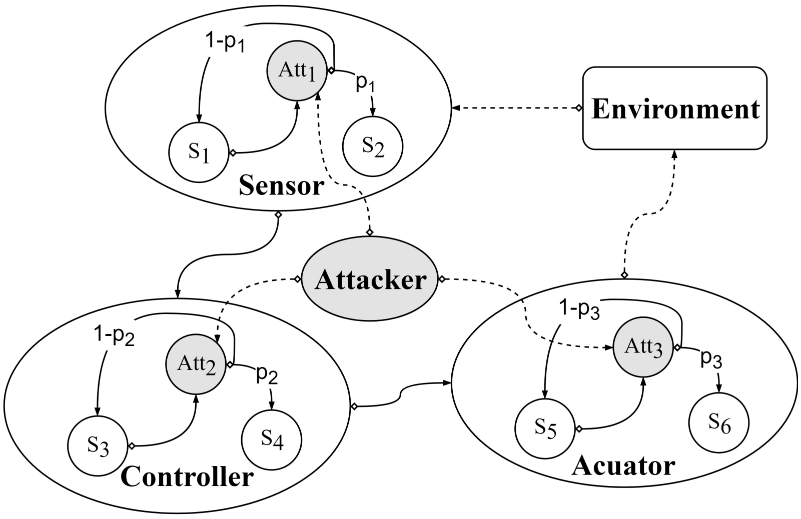


Figure 4. MDP model for security risk analysis.

Finally, our MDP model is formalized in the PRISM language. The basic elements of PRISM include modules and variables, and a model is usually composed of various modules. Variables describe the state of each module at any given time. The state transition of each module is described by several columns of commands. A command takes the following format:

$$[]guard \rightarrow prob_1 : update_1 + ... + prob_n : update_n;$$

where *guard* is a predicate over all the variables, each $update_i$ describes a transition that the module can make if the guard is true, and each update is also assigned a probability $prob_i$ that satisfies $\sum_{i=1}^{n} prob_i = 1$. Each module and its signals in our MDP model are modeled as a module and internal variables of the PRISM language, respectively.

4.4. Security Property Specification and Evaluation

In order to analyze a given MDP model that has been specified and constructed in the PRISM language, it is necessary to define the security properties of the target system (i.e., the goal of analysis). As mentioned above, the controller defines many policies for dynamic control of in-vehicle systems, such as reducing engine speed when overspeed is detected. In addition to system failures caused by system functional problems, we do not want network security problems to cause behaviors inconsistent with predefined policies. Therefore, we always assume that every component of the on-board system always meets the functional safety requirements of ISO 26262. Our description of security properties is to define all possible inconsistent behaviors and then analyze the possibility of these inconsistent behaviors. In our model, we are concerned about the maximum and minimum probability of these inconsistent behaviors. PRISM provides two possible property types, $Pmax$ and $Pmin$, which compute the maximum and minimum probability values of the state formula, respectively. Path properties that can be used in $Pmax$ and $Pmin$ operators include X (next), U (until), F (future), G (globally), W (week until), and R (release). To calculate the values for all states simultaneously, we can choose to use the $filter$ operator, which takes the following form:

$$filter(min, P = ?[pathprop], states)$$

$$filter(max, P = ?[pathprop], states)$$

5. Illustration

In this section, the proposed framework is applied to the actual automotive collision warning system, which is a typical safety-critical system composed of various sensors, controllers, actuators, networks, and other components. As a contrast, we analyze the security risk of the system under the traditional distributed and the modern centralized in-vehicle architecture. In addition, as an assistant method, a model transformation algorithm is introduced to generate MDP expressions from a SysML activity diagram.

5.1. Target of Evaluation

Figure 5 shows a modern centralized in-vehicle electronic and electrical architecture. The automotive electronic system is divided into several domains according to their functions. Among them, the ADAS controller is used to realize environmental perception and decision. The power domain is used for engine control, such as acceleration and deceleration. The chassis domain is used to achieve chassis control, such as steering control. The infotainment system includes instrument clusters, audio-visual entertainment, and warning lights. The network in the vehicle is connected to the external network through the connected gateway for remote diagnosis, remote update, and other purposes. The automobile collision system can be regarded as a subsystem of the ADAS system, including a lane departure warning system (LDWS) and a forward collision warning (FCW) system. The sensors involved include ultrasonic sensors and camera sensors, which are respectively used to sense if vehicles get too close and whether the vehicle begins to move out of its lane. The actuator includes an engine, a steering wheel, and a warning light. An ADAS controller generates a control signal according to sensor signals, and the control signals are sent to actuators through various controllers and links. The functions provided by the automotive collision warning system can be described in detail as follows:

- *Function 1.* When the ultrasonic sensor S_1 detects that the vehicle ahead is too close, the ADAS controller C_1 sends a deceleration command to the engine controller A_1 and a lock command to steering controller A_2.
- *Function 2.* When the camera sensor detects that the car is about to move out of its lane, the ADAS controller C_1 sends an unlocking command to the steering wheel controller A_2 and sends a warning signal to the alarm controller A_3 to remind the driver to adjust the steering wheel.
- *Function 3.* The gateway controller C_2 is connected to the Internet through the cellular network for remote diagnosis and firmware refreshing.
- *Function 4.* The Head unit controller C_6 is connected to the external devices and services, such as smartphones and cloud music.
- *Function 5.* The connected gateway is connected to RSU D_1 and cloud server D_2. Basic safety messages (BSM) can be received from D_1 through DSRC communication. The cloud server D_2 provides remote diagnosis functions, one of which is querying the status of the alarm A_3.

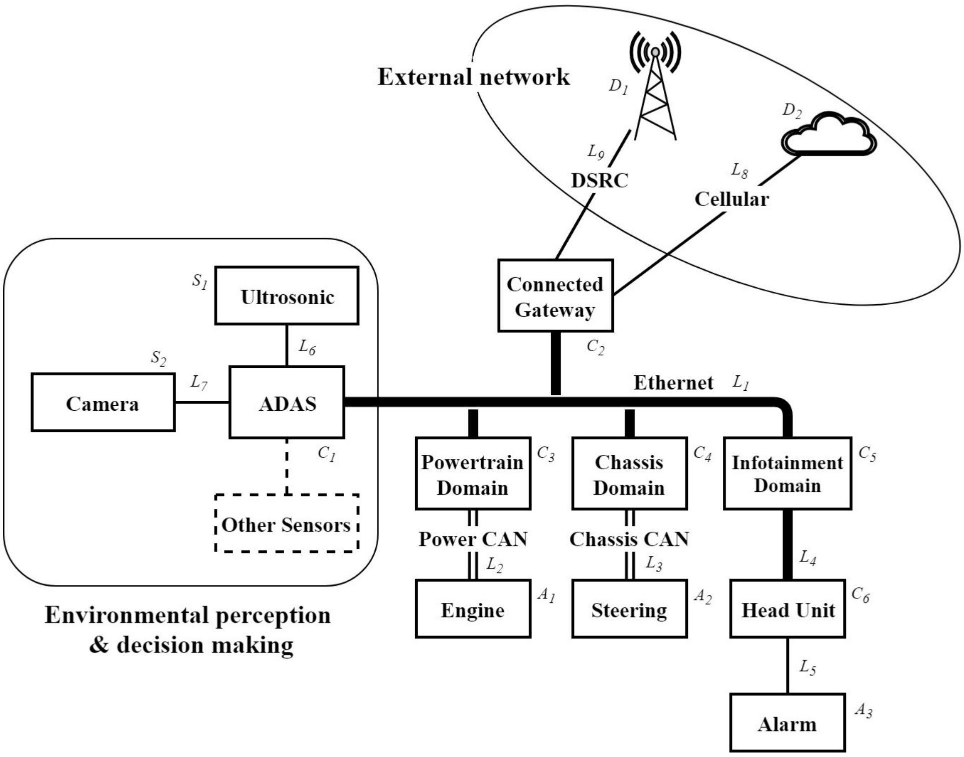


Figure 5. Collision warning system based on centralized EEA.

5.2. Threat Identification

Threat identification is used to identify possible security threats inside each target of evaluation (TOE) module, such as message tampering, malicious code injection, and message blocking. Attack surfaces and corresponding threats against connected vehicles have been investigated by [4,11,16,26]. In our example, the possible attacks were extracted from these research efforts. Petit et al. [4] performed further work to classify the identified attacks against the connected vehicles into different levels based on the probability of success. For our system, the assets and the attack methods they face are shown in Table 3, and each atomic attack is labeled att_i. The probability and impact of each attack are also listed in the table (after normalization).

Table 3. Identified assets and threats.

Labels	Assets	Attack Methods	Probability	Normalized Probability	Impact Level
att_0	Connected gateway	Malware injection	Almost Certain	75%	4
att_1	Head unit	Head unit attack	Chances About Even	50%	2
att_2	ADAS controller	Manipulation	Chances About Even	50%	4
att_3	Ultrasonic sensor	Blind	Probably Not	30%	3
att_4	Camera sensor	Blind	Probably Not	30%	3
att_5	Engine	Inject CAN message	Almost Certainly Not	7%	4
att_6	Steering	Inject CAN message	Almost Certainly Not	7%	4
att_7	RSU	Inject message	Almost Certain	75%	1

5.3. Model Generation

As mentioned above, our MDP model is composed of a system model, identified threats, and attacker model. For the purpose of reusability, we modeled the system model in the PRISM tool as four types of modules: an environment module, a sensor module, a controller module, and an actuator module. The environment module describes the uncertainty of the environment's change, in our case representing the change of the vehicle's distance and lane line. The sensor module describes the process of sensors perceiving environment changes. The controller module describes the process of how the output signal is generated according to the input signal by the controller. The actuator module describes the process in which the actuator performs corresponding actions according to the received control signals. To understand the behavior within each module, Figure 6 shows two example statements in the ADAS controller module. Variable *att_mani* = 0 indicates that the manipulation attack did not occur. In this case, the ADAS controller will generate control signals according to the ultrasonic sensor signal *dist_val* and *lane_val* to control the action of the engine, steering wheel, and alarm. By contrast, *att_mani* = *1* indicates that a manipulation attack occured, and the ADAS controller will not generate correct control signals.

```
[] att_mani=0 & dist_val=1 & dist_val_flag=1 ->
(comm_brake'=1) & (comm_alarm'=1) & (alalrm_flag'=1) &
(brake_flag'=1);
[] att_mani=1 & dist_val=1 & dist_val_flag=1 ->
(comm_brake'=0) & (comm_alarm'=0) & (alalrm_flag'=1) &
(brake_flag'=1);
```

Figure 6. Example statements in ADAS controller module.

In each module, a boolean value is used to indicate whether an attack occurs (e.g., *att_mani*), which is determined by the attacker. As the behavior of the attacker is non-deterministic, the maximum rewards of the attacker are limited by the attacker's capability. In this case, we built a module to describe the attacker's behavior. In order to understand the uncertain behavior of attackers, Figure 7 shows two example statements in the attacker module. The first statement describes that the probability of success of malware injection is 0.75 and the probability of failure is 0.25. If the atomic attack succeeds, the attacker's reward *reward* will be increased by 4 according to the impact level; otherwise, the reward *reward* will not be increased. It is worth noting that the attacker's reward can never exceed the attacker's capability *att_cap*. Similarly, the second statement describes that the probability of success of the head unit attack is 0.5. Other attacker behaviors can be created in a similar way.

```
[] att_mal=0 & net3_flag=1 & reward<att_cap &
counter=1 -> 0.75:(att_mal'=1) & (reward'=reward+4) &
(counter'=2) + 0.25:(att_mal'=0) & (reward'=reward) &
(counter'=2);
[] att_head=0 & con_head_flag=1 & reward<att_cap &
counter=2 -> 0.5:(att_head'=1) & (reward'=reward+2) &
(counter'=3) + 0.5:(att_head'=0) & (reward'=reward) &
(counter'=3);
```

Figure 7. Example statements in the attacker module.

As a comparative analysis, the collision warning system under the traditional distributed EEA is shown in Figure 8. The MDP model under this architecture can be built in a similar way to the one mentioned above. Under these two architectures, the description of security properties is universal, and the description and evaluation of security properties will be described in the next subsection.

5.4. Properties Specification and Evaluation

Here, we present six security properties to be verified against the TOE shown in Figure 5. Each security property is first described in natural language, and then is translated into a formal description by using PCTL, which can be used as the input of the PRISM tool. After verification, we can get the attacker's maximum probability to break each property. For easy understanding of property descriptions, some of the variables involved in this use case are shown in Table 4.

Property 1. "Calculate the maximum probability that the alarm will not respond properly". Whether the alarm works normally can be divided into two situations: one is that no control signal is received, and the other is that the received control signal was tampered with. A formal description in PRISM language of *Property 1* is $filter(max, Pmax = \ ?, [G(alarm = 0|alarm = 2)\&reward < x], dist = 1)$, where x is a variable that represents the attacker's capability.

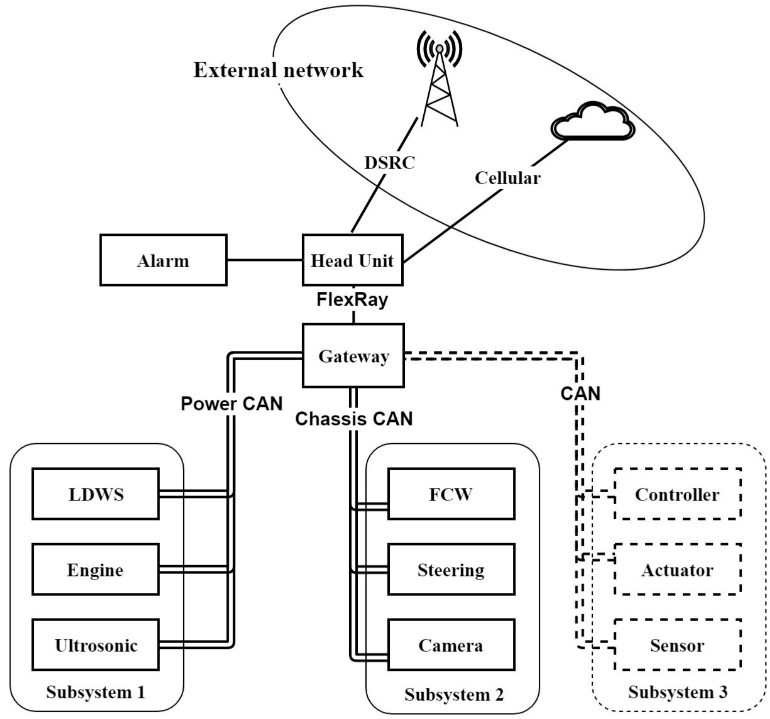


Figure 8. Collision warning system based on distributed EEA.

Table 4. Description of some of the variables involved in the example.

Comments **Possible Values** **Variables**	**0**	**1**	**2**
lane	In the lane	Out of the lane	╲
dist	Normal distance	Abnormal distance	╲
safety	No safety event	Safety event	╲
alarm	Alarm should not be triggered	Alarm should be triggered	No signal
dist_val	Normal distance is detected	Abnormal distance is detected	No signal
bsm	Safety event is not detected	Safety event is detected	No signal
comm_steer	Command to lock steering wheel	Command to unlock steering wheel	No signal
diag_alarm	Alarm is not be triggered	Alarm is triggered	No signal

Property 2. "Calculate the maximum probability that an ultrasonic sensor signal is wrong". Sensor data errors can be divided into three situations: sensor blinding, data manipulation, and denial of service. A formal description in PRISM language of *Property 2* is $filter(max, Pmax = ?, [G(dist\_val = 0|dist\_val = 2)\&reward < x], dist = 1)$, where x is a variable that represents the attacker's capability.

Property 3. "Calculate the maximum probability that a steering control signal is wrong". The steering control signal is generated by the ADAS controller according to the predefined strategy, which is affected by the correctness of the sensor data, the security of the gateway and the ADAS controller, and the security of the CAN bus in the vehicle. A formal description in PRISM language of *Property 3* is $filter(max, Pmax = ?, [G(comm\_steer = 0|comm\_steer = 2)\&reward < x], lane = 1)$, where x is a variable that represents the attacker's capability.

Property 4. "Calculate the maximum probability that a remote diagnosis signal from the alarm to the cloud is wrong". Sometimes remote cloud servers want to know if the alarm is triggered, and this information is usually sent from the local vehicle to the cloud through a diagnostic message. This information is likely to be tampered with in the process of transmission. A formal description in PRISM language of *Property 4* is $filter(max, Pmax = ?, [G(diag\_alarm = 0)\&reward < x], lane = 1|dist = 1)$, where x is a variable that represents the attacker's capability.

Property 5. "Calculate the maximum probability that the alarm causes a wrong response to a safety event". When the vehicle is driving, infrastructure (such as RSU) usually reports the surrounding environment information to the vehicle through DSRC communication. A basic safety message (BSM) is an example of this information and wrong BSM can cause s life-threatening situation to drivers, passengers, and surrounding vehicles. A formal description in PRISM language of *Property 5* is $filter(max, Pmax = ?, [G(alarm = 0|alarm = 2)\&reward < x], safety = 1)$, where x is a variable that represents the attacker's capability.

Property 6. "Calculate the maximum probability of denial of service". In this particular case, a DoS means that all the signals we care about are in a "no signal" state. A formal description in PRISM language of *Property 6* is $filter(max, Pmax = ?, [G(alarm = 2\&dist\_val = 2\&bsm = 2\&comm\_steer = 2\&diag\_alarm = 2)\&reward < x], 1)$, where x is a variable that represents the attacker's capability.

As an illustration only, we have not considered all of the security properties of the TOE. In a practical use case, we may define more security properties for verification according to our proposed method. After verification, as the attacker's capability grows, the changes of the maximum probability of breaking each property under both EEAs are shown in Figure 9. Up to now, we have realized the quantitative security risk analysis for a safety-critical system of connected vehicles. The results of the evaluation show that the risk of each security property is related to the capacity of the attacker and system architecture, and more discussion will be presented later. Another noteworthy result is that atomic attacks are exploited when the risk of attack reaches the maximum under the limited attacker's ability. For each property is this case, when the attacker's ability is 12, the corresponding atomic attacks are shown in Table 5.

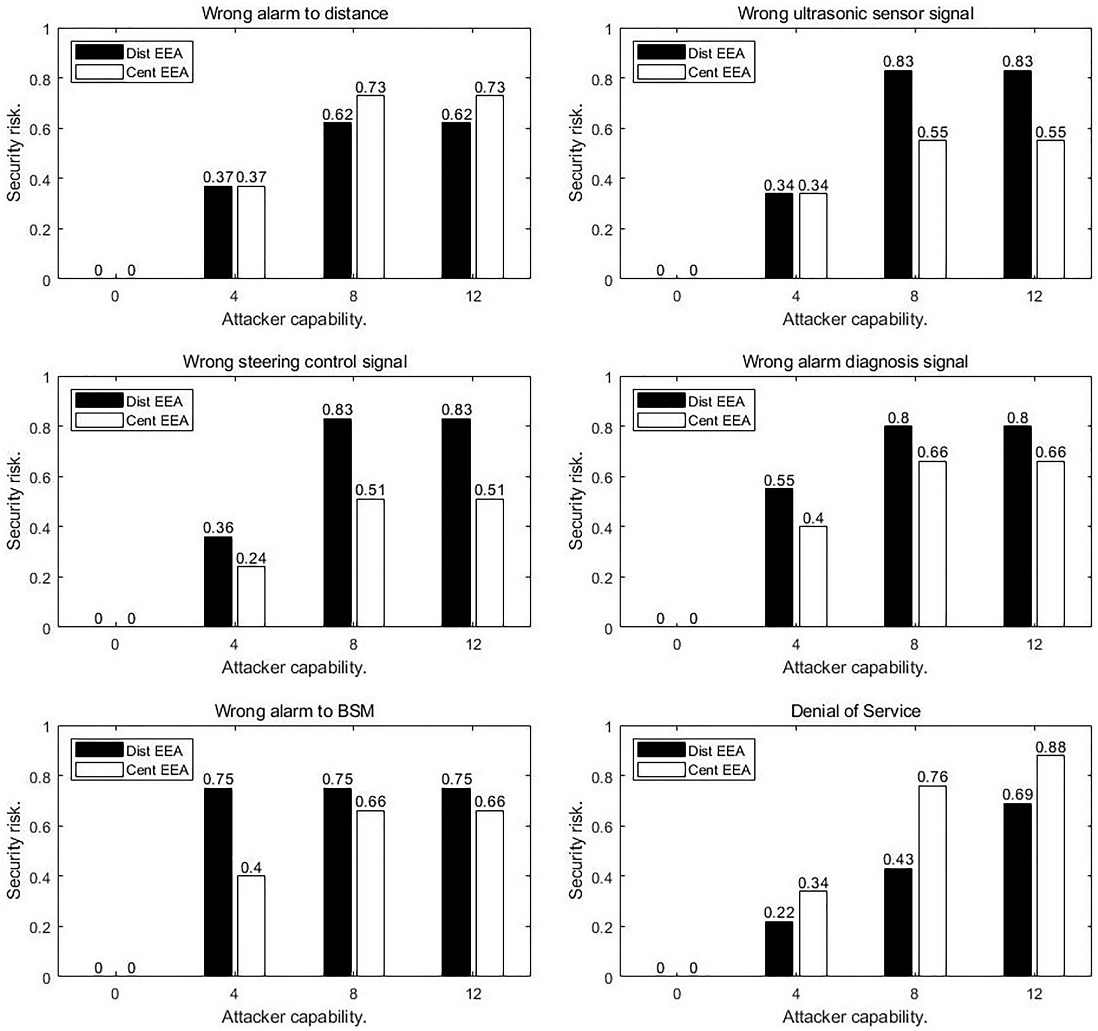


Figure 9. Evaluation results.

Table 5. Atomic attacks when attacker capability is 12.

Properties	Atomic Attacks in Distributed EEA	Atomic Attacks in Centralized EEA
Property 1	att_0, att_5	att_0, att_1, att_3
Property 2	att_0, att_3, att_5	att_2, att_3
Property 3	att_0, att_1, att_6	att_0, att_2
Property 4	$att_0, att_1, att_5, att_6$	att_0, att_1, att_2
Property 5	att_7	att_0, att_1, att_2
Property 6	*all*	*all*

5.5. Implementation

Most of the work in the security risk assessment approach is to establish the MDP model. Automotive engineers are more familiar with the electrical and electronic architecture of the vehicle and are not good at model checking technology. Therefore, they prefer to focus on the description of the automotive electronic and electrical systems and their functions, while the MDP model generation work is handed over to the computer to complete. Therefore, we propose an engineering method to generate the MDP model for safety-critical systems of connected vehicles by using the SysML activity diagram. Firstly, we use the SysML activity diagram to describe our TOE's structure and functions, and selected activity diagram elements are mapped to components and state transitions of TOE in the diagram. For example, $Partition$ is used to describe modules, $ControlFlow$ is used to describe message passing, $Action$ is used to describe atomic activities inside each module, and so on. Then, we use the Python API to implement a model generation tool, which takes the system model file described by XML as input, and the output results are MDP expressions described with the PRISM language. Finally, the generated MDP expressions can be reorganized together and imported into the PRISM tool for verification. The MDP expressions generation algorithm based on depth first search (DFS) is given in Algorithm 1.

Algorithm 1: SysML Activity Diagram Transformation Algorithm.

```
Input: SysML activity diagram XML file
Output: MDP expressions in PRISM language
/* Initialization */
nodes as Stack;
cNode as Node;
mdpExpress as list_of_Expression;
/* Push initial node in the stack. */
nodes.push(init);
for all n in init do
    nodes.push(n)
end
while not nodes.empty() do
    /* Pop the current node. */
    cNode := nodes.pop();
    if cNode is not visited then
        /* Current node has been visited.*/
        visited(cNode);
        /* Call the mapping rules function */
        mdpExpress.push(MDPexp.gen(init, cNode));
    end
    /* Clear visited flag. */
    clear(cNode);
end
```

In Algorithm 1, *nodes* is the activity diagram's nodes stored in the stack, *cNode* is the node being accessed, and *mdpExpress* is the set of generated MDP expressions. The *MDPexp.gen()* function converts the currently accessed SysML activity diagram elements to an MDP expression. The mapping rules between MDP expression and SysML activity diagram constructs are shown in Table A1. The transition algorithm calls DFS at most once for each node in the activity diagram and searches the adjacency matrix every time it calls DFS. Therefore, the complexity of the algorithm is $O(n^2)$, where n represents the number of nodes in the activity diagram.

5.6. Discussion

In-vehicle embedded systems are becoming more and more complex, due to emerging connectivity and intelligent driving. As a result, it is becoming hard to analyze the security risk of the safety-critical systems inside connected vehicles. Automotive engineers have difficulty understanding the security threat of in-vehicle safety-critical systems by relying on traditional threat analysis methods. Our proposed approach can realize the system-level analysis of the security risk of the in-vehicle safety-critical system by formalizing a system model and a threat model. This formal method can avoid the ambiguity of the system security properties' definition, which is very important for safety-critical systems of connected vehicles. The key findings from our illustration are summarized here:

- For the same safety-critical system, the security risks faced by the system are very different under different automotive electronic and electrical architectures (as shown in Figure 9).
- In a traditional distributed architecture, sensor signals and control signals are transmitted in a CAN bus through broadcast. Moreover, the CAN bus is directly connected to the in-vehicle gateway, and these two signals have a probability of being successfully attacked of up to 0.83. In the centralized architecture, the CAN bus in the vehicle is isolated, so the attacker cannot penetrate the CAN bus, and thus the probability of being attacked is lower, at about 0.5.
- Under the modern centralized electronic and electrical architecture, the automobile provides more external interfaces (i.e., connected gateway and head unit). Attackers can have more attack methods against alarm signals. Therefore, the probability of success of attacking an alarm signal in the centralized architecture is 0.73, higher than that in a distributed architecture.
- When the vehicle communicates with the outside world, such as remote diagnosis and basic safety messages (BSM), the risk level of the traditional distributed architecture (at 0.8, and 0.75 respectively) is greater than that of the centralized architecture (at 0.6), because of the simple domain isolation mechanism.
- Because the centralized architecture adopts an IP-based backbone network, this open network system is more vulnerable to denial of service (DoS) attacks.
- In our proposed approach, due to having discretized and abstracted the sensor signal and control signal, we must know the internal strategy of the system. This kind of abstraction may be very different in different engineers' concrete practices.

It is worth noting that because we aimed to test our implementation in an environment that is as realistic as possible, noise was inserted into the scenario. From the perspective of practice, model noise can be removed from three aspects: (1) threats in the actual work may come from different organizations and systems, which should be credible and categorized according to different modules and organizations; (2) a unified scoring system, similar to a common vulnerability scoring system (CVSS) and common vulnerabilities and exposures (CVE), should be adopted in the analysis process to assess the security risk of threats; and (3) security threat analysis conducted by a security engineer should be based on a system engineer's description of the target system, and as many security engineers as possible should be involved in order to reduce the error.

6. Conclusions

Connected and automatic vehicles are facing a number of challenges in security threats. Avoiding the risks caused by cyberattacks against automotive safety-critical systems is very important for ensuring the safety of drivers and passengers. In this paper, we have proposed a security risk assessment framework for the safety-critical system of the connected vehicles. A formal method is adopted to build a system model and threat model of the TOE, which can describe the in-vehicle EEA and its security properties more accurately. The results delivered by our approach can help automotive security engineers have a system-level understanding of the TOE's security risks at the early stage of security design. The case study in this paper demonstrated that our proposed approach is easy to use and effective. In addition, the qualitative-to-quantitative transformation method proposed in this paper makes the framework compatible with some previous frameworks, such as EVITA, HEAVENS, SAHARA and FMVEA. Automated model generation tools can greatly improve the efficiency of threat modeling and thus reduce the time cost of security risk assessment.

In the future, we plan to extend our framework to enable it to handle scenarios where multiple threats exist on a single module. Another possible task is to benchmark our approach with security assessment standards in the automotive sector, such as ISO/SAE 21434 (to be released). Once the benchmarking work is completed, we will be able to carry out threat analysis for realistic application scenarios and establish a rich threat database, which is meaningful but rather complex work. In addition, we will improve the toolchain to make it more automatic, so that non-experts can also use this toolchain to finish valuable security risk analysis and verification work of the safety-critical system inside connected vehicles.

Author Contributions: Conceptualization, F.L.; formal analysis, S.H. and X.Z.; methodology, S.H.; software, S.H. and X.Z.; supervision, F.L.; validation, S.H. and X.Z.; writing—original draft, S.H.; writing—review and editing, Z.Y. and W.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Shanghai Science and Technology Commission (grant number 18DZ1101300), Shanghai, China.

Conflicts of Interest: The authors declare no conflict of interest.

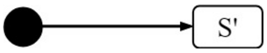
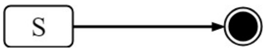
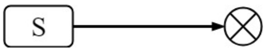
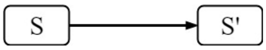
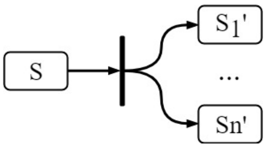
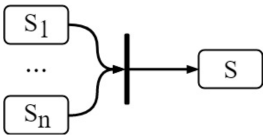
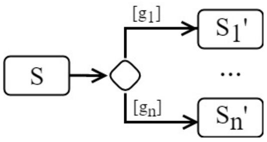
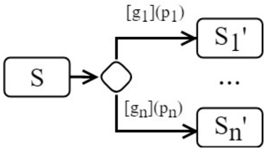
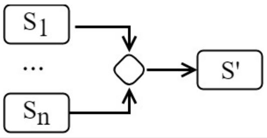
Abbreviations

The following abbreviations are used in this manuscript:

EEA	Electrical and Electronic Architecture
OCTAVE	Operationally Critical Threat, Asset, and Vulnerability Evaluation
EVITA	E-safety vehicle intrusion protected applications
HEAVENS	HEAling Vulnerabilities to ENhance Software Security and Safety
SAHARA	Security-Aware Hazard and Risk Analysis
ATA	Attack Tree Analysis
FMVEA	Failure Mode, Vulnerabilities, and Effects Analysis
IDS	Intrusion Detection System
TOE	Target of Evaluation
ECU	Electronic Controller Units
ADAS	Advanced Driving Assistance System
LDWS	Lane Departure Warning System
FCW	Forward collision warning
MDP	Markov Decision Process
PCTL	Probabilistic Computation Tree Logic
TARA	Threat Analysis and Risk Assessment
ASIL	Automotive Safety Integration Level
SysML	System Modeling Language
DFS	Depth First Search

Appendix A

Table A1. Mapping rules between MDP expressions and SysML activity diagram constructs.

Activity Diagram Constructs	MDP Expressions	Descriptions
	$ini \rightarrow s^{'}$	Activity initial node.
	$s \rightarrow fin$	Activity final node.
	$s \rightarrow FlowFinal$	Activity flow final node.
	$s \rightarrow s^{'}$	Action node.
	$s \rightarrow (s_1^{'} \wedge s_2^{'} \wedge ... \wedge s_n^{'})$	Fork node.
	$(s_1 \wedge s_2 \wedge ... \wedge s_n) \rightarrow s^{'}$	Join node.
	$s \rightarrow [g_1]s_1^{'} \vee [g_2]s_2^{'} \vee ... \vee [g_n]s_n^{'}$	Branch node.
	$s \rightarrow [g_1](p_1)s_1^{'} \vee [g_2](p_2)s_2^{'} \vee ... \vee [g_n](p_n)s_n^{'}$	Probabilistic branch node.
	$(s_1 \vee s_2 \vee ... \vee s_n) \rightarrow s^{'}$	Merge node.

References

1. Hillenbrand, M.; Heinz, M.; Matheis, J.; Müller-Glaser, K.D. Development of electric/electronic architectures for safety-related vehicle functions. *Softw. Pract. Exp.* **2012**, *42*, 817–851. [CrossRef]
2. Mundhenk, P.; Steinhorst, S.; Lukasiewycz, M.; Fahmy, S.A.; Chakraborty, S. Security analysis of automotive architectures using probabilistic model checking. In Proceedings of the 52nd Annual Design Automation Conference (DAC'15), San Francisco, CA, USA, 8–12 June 2015; pp.1–6.
3. Koscher, K.; Czeskis, A.; Roesner, F.; Patel, S.; Kohno, T.; Checkoway, S.; McCoy, D.; Kantor, B.; Anderson, D.; Shacham, H.; et al. Experimental Security Analysis of a Modern Automobile. In Proceedings of the 31st IEEE Symposium on Security & Privacy, Oakland, CA, USA, 16–19 May 2010; pp.447–462.
4. Petit, J.; Shladover, S.E. Potential cyberattacks on automated vehicles. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 546–556. [CrossRef]

5. Zhang, Y.; Ge, B.; Li, X.; Shi, B.; Li, B. Controlling a Car Through OBD Injection. In Proceedings of the IEEE 3rd International Conference on Cyber Security and Cloud Computing (CSCloud 2016), Beijing, China, 25–27 June 2016; pp.1–4.
6. Macher, G.; Eric, A.; Eugen, B.; Christian, K. A Review of Threat Analysis and Risk Assessment Methods in the Automotive Context. In Proceedings of the Computer Safety, Reliability, and Security (SAFECOMP 2016), Trondheim, Norway, 21–23 September 2016; pp.130–141.
7. SAE International. *Cybersecurity Guidebook for Cyber-Physical Vehicle Systems*; SAE International: Warrendale, PA, USA, 2016; pp. 70–90._201601. [CrossRef]
8. Woo, S.; Jo, H.J.; Lee, D.H. A Practical Wireless Attack on the Connected Car and Security Protocol for In-Vehicle CAN. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 993–1006._11. [CrossRef]
9. Takahashi, J. An Overview of Cyber Security for Connected Vehicles. *IEICE Trans. Inf. Syst.* **2018**, *101*, 2561–2575. [CrossRef]
10. Wolf, M.; Gendrullis, T. Design, implementation, and evaluation of a vehicular hardware security module. In Proceedings of the 14th international conference on Information Security and Cryptology (ICISC 2011), Seoul, Korea, 30 November–2 December 2011; pp. 302–318.
11. Mohammad, N.; Muhammad, S.; Shaikh, E. Analysis of In-vehicle Security System of Smart Vehicles. In Proceedings of the 5th International Conference on Future Network Systems and Security (FNSS 2019), Melbourne, VIC, Australia, 27–29 November 2019; pp. 198–211.
12. Mukherjee, S.; Shirazi, H.; Ray, I.; Daily, J.; Gamble, R. Practical DoS Attacks on Embedded Networks in Commercial Vehicles. In Proceedings of the 12th International Conference on Information Systems Security (ICISS 2016), Jaipur, India, 16–20 December 2016; pp. 23–42.
13. Bae, W.S. Function-based connection protocol development and verification for secure communication in vehicle environment. *Cluster Comput.* **2015**, *18*, 761–769. [CrossRef]
14. Luo, F.; Hou, S. *Security Mechanisms Design of Automotive Gateway Firewall*; SAE Technical Paper 2019-01-0481; SAE International: Warrendale, PA, USA, 2019; pp.1–8. [CrossRef]
15. Al-Jarrah, O.Y.; Maple, C.; Dianati, M.; Oxtoby, D.; Mouzakitis, A. Intrusion Detection Systems for Intra-Vehicle Networks: A Review. *IEEE Access* **2019**, *7*, 21266–21289. [CrossRef]
16. Ruddle, A.R.; Ward, D.D. *Intelligent Transport Systems: Technologies and Applications*; Wiley-Blackwell: Hoboken, NJ, USA, 2015; pp. 83–106, ISBN 978-1-118-89477-4.
17. Automotive Security and Privacy: Holistic Approach to Improve Data Security. Available online: https://autosec.se/wp-content/uploads/2018/03/HOLISEC_D4.1.1_v1.0.pdf (accessed on 6 May 2020).
18. Kadhirvelan, S.P.; Söderberg-Rivkin, A. Threat Modelling and Risk Assessment Within Vehicular Systems. Master's Thesis, Chalmers University of Technology, Gothenburg, Sweden, 2014; pp. 16–19.
19. Wang, P.; Wu, X.; He, X. Modeling and analyzing cyberattack effects on connected automated vehicular platoons. *Transp. Res. Part C Emerg. Technol.* **2020**, *115*, 102625. [CrossRef]
20. Hamad, M.; Prevelakis, V. SAVTA: A hybrid vehicular threat model: Overview and case study. *Information* **2020**, *11*, 273. [CrossRef]
21. Carreras Guzman, N.H.; Wied, M.; Kozine, I.; Lundteigen, M.A. Conceptualizing the key features of cyber-physical systems in a multi-layered representation for safety and security analysis. *Syst. Eng.* **2020**, *23*, 189–210. [CrossRef]
22. Kong, H.-K.; Hong, M.K.; Kim, T.-S. Security risk assessment framework for smart car using the attack tree analysis. *J. Ambient Intell. Hum. Comput.* **2018**, *9*, 531–551. [CrossRef]
23. Macher, G.; Sporer, H.; Berlach, R.; Armengaud, E.; Kreiner, C. SAHARA: A security-aware hazard and risk analysis method. In Proceedings of the 2015 Design, Automation Test in Europe Conference Exhibition (DATE 2015), Grenoble, France, 9–13 March 2015; pp. 621–624.
24. Mohsin, M.; Sardar, M.U.; Hasan, O.; Anwar, Z. IoTRiskAnalyzer: A Probabilistic Model Checking Based Framework for Formal Risk Analytics of the Internet of Things. *IEEE Access* **2017**, *5*, 5494–5505. [CrossRef]

25. Ouchani, S.; Ait Mohamed, O.; Debbabi, M. A Security Risk Assessment Framework for SysML Activity Diagrams. In Proceedings of the 2013 IEEE 7th International Conference on Software Security and Reliability (SERE 2013), Gaithersburg, MD, USA, 18–20 June 2013; pp. 227–236.
26. Kawanishi, Y.; Nishihara, H.; Souma, D.; Yoshida, H.; Hata, Y. A Comparative Study of JASO TP15002-Based Security Risk Assessment Methods for Connected Vehicle System Design. *Secur. Commun. Netw.* **2019**, *2019*, 4614721. [CrossRef]