ENABLING A MULTI-PRONGED SOCIO-TECHNICAL APPROACH TO ADDRESS AUTOMOTIVE CYBERSECURITY

By

Nicholas Polanco

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science—Doctor of Philosophy

2025

ABSTRACT

The increase of inward-facing and outward-facing communication used by modern vehicles with automated features expands the breadth and depth of automotive cybersecurity vulnerabilities. The prominent role that human behavior plays in the lifetime of a vehicle creates a need for social and human-based factors to be considered in tandem with the technical factors when addressing cybersecurity. Specifically, in collaboration with researchers from criminology and the automotive industry, we integrate foundations of crime theory, human factors, and model-driven engineering to develop three complementary automotive cybersecurity prevention strategies, where they differ in the balance between technical and social emphasis, both in terms of solution strategies and the targeted stakeholders. We start with a stakeholder-aware threat assessment to analyze automotive systems for vulnerabilities and solutions across the spectrum of stakeholders using the vehicle. In addition to identifying attack surfaces, this threat assessment includes relevant human-focused information, such as type of access needed by attacker (e.g., physical or remote; time needed to complete attack), attacker background knowledge, and impact on human safety. This threat assessment is used to inform all three of our approaches to automotive cybersecurity. First, we developed a set of technical automotive cybersecurity design patterns (i.e., reusable designs for specific cybersecurity problems), targeting the technical stakeholder group. Second, leveraging situational crime prevention strategies, we developed a configurable situational crime prevention framework for automotive cybersecurity where we consider both state of the art and state of the practice strategies to address vulnerabilities. Targeted stakeholders include dealerships, OEM's, and developers. Finally, we developed socio-technical design patterns that provide reusable solution strategies that engage the broader community to address automotive cybersecurity. Example stakeholders include third party vendors, automotive hobbyists, and white-hat attackers. These three strategies provide reusable solutions to be realized by a spectrum of technical and social-based activities to address automotive cybersecurity, which engages the broader community, thus increasing the overall impact and societal benefits. This dissertation takes an interdisciplinary approach to address automotive cybersecurity where we synergistically combine cybercrime theory,

human factors, and technical solutions to develop reusable prevention and detection techniques.

ACKNOWLEDGMENTS

This small section of text does not do justice to the amount of help and support I was given from so many people, but I will do my best to extend my gratitude properly. I would first like to thank Dr. Betty H. C. Cheng for her assistance from my earliest moments at Michigan State University. Despite the number of obstacles I encountered, either through graduate studies or in my personal life, she always gave guidance and support. When all my energy was spent, I could count on her to keep my spirits high and help me reach deadlines. The numerous pages of feedback, revisions, and red text on papers and projects will always be a reminder to keep working. Also, a special thanks to Dr. Philip McKinley who has continued to assist me with my professional career. Thank you for the numerous meetings, either via Zoom or in your office, and guiding me through my first years at Michigan State. You were always willing to meet with me, and offered honest and thorough feedback every time. I also would like to thank my committee members: Dr. Josh Siegel, Dr. Thomas Holt, and Dr. Kristen Johnson. Their contributions have been greatly appreciated and helped provide insight into the work that was being done. The members of the SENS lab such as Glen Simon, Kira Chan, and Sol Zilberman always provided valuable conversation on our various projects, as well as being good friends. The other people outside of the college I would like to thank are the great friends in my life like Charles Weber, Mitch Regel, and Gustavo Bueno. The love, support, and motivation has always helped me reach my goals despite feeling exhausted, thank you. Also, a special thanks to Savannah Rana and her unyielding support and love. Finally, I would like to thank my entire family for believing in me, and never having any doubt. I know that you all suffered with me through this long process, but I could always count on you to have high spirits and be my pillar of strength. Thank you for everything you have done for me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The expansion of communication, both inward-facing and outward-facing, used by self-driving vehicles has revealed numerous automotive cybersecurity vulnerabilities. Given that software-based errors in self-driving vehicles can cause life threatening consequences [8], it becomes necessary to address automotive cybersecurity vulnerabilities as we move closer to a broad deployment of these vehicles [9]. The autonomous vehicles have many users interacting with different elements of the system. As such, the vulnerabilities and the sources of malicious behavior are broad [10], posing significant challenges when developing defense strategies. Furthermore, the need for a secure system operating in a constantly-changing environment is critical to the safety of humans inside and outside the vehicle.

To date, research related to securing modern cars has been largely software-based and technical in nature [11, 12, 13]. Findings outlined in a number of survey papers and research literature, the technical vulnerabilities in these self-driving cars are being addressed and solutions are being proposed in preparation of the release of these vehicles. When considering software errors handled by development teams, cybersecurity attacks can also be prevented by leveraging different stakeholders across the development and maintenance of these vehicles. As such, a socio-technical solution is needed to complement and/or supplement this challenging problem. Although much of the technical security vulnerabilities within these systems are being addressed, a more proactive and preventative mindset that explicitly considers the human factors and influencing behavior inside the automotive space is needed to provide additional support to vehicle security.

1.1 Problem Description

The complexity and importance of securing autonomous vehicles is immense, where the focus has predominantly been on software bugs, vulnerabilities, anti-virus, and other technical issues. However, the communication systems used inside these vehicles make it necessary to account for a range of users interacting with the system. Securing self-driving vehicles extends beyond the technical and software-based approaches. The different stakeholders interacting with these vehicles at various stages also allows attackers different path ways of infiltrating a system. In order to more comprehensively protect the vehicles, these human-centered interactions need to be explicitly acknowledged and addressed. The main goal of our work is to focus on the different types of actions of stakeholders and provide reusable solutions to secure these vehicles.

1.2 Thesis Statement

This research is focused on the stakeholders using the systems, as well as their range of behavior and interactions to collectively address automotive cybersecurity. The stakeholders, ranging from users to developers, requires a more human-aware strategy for cybersecurity. The implementation and organization of these strategies allow for a more robust, secure system. Specifically, we introduce complementary socio-technical solutions that combine both criminology principles and common software engineering practices.

Thesis Statement. *A socio-technical framework can be used to integrate technical, social, and socio-technical approaches to develop reusable solution strategies for addressing cybersecurity vulnerabilities in automotive systems.*

This doctoral research focuses on defending these vehicles against cyber attacks in a preemptive, socio-technical fashion by securing different vulnerabilities based on the relevant stakeholders interacting with the system. The stakeholders include the people responsible for manufacturing, the developers responsible for software issues, technicians working on repairs, and customers who own the vehicles. After we identify these contributing stakeholders, we generalize solution strategies to apply at a high-level, for multiple vulnerabilities that impact the automotive systems. The strategy is achieved by synergistically combining both technical and social solutions within this unique setting of autonomous vehicles. We can use a specialized, stakeholder-focused threat assessment to identify a spectrum of stakeholders who may experience an attack during the lifecycle of the vehicle. The focus on stakeholders provides a means of analyzing different facets of how an attacker penetrates an automotive system by identifying knowledge needed, time required, and distance to attack the automotive system. The technical-focused stakeholders are provided with automotive security design patterns focused on reusable problem solution sets. The integration of crime theory concepts enables the development of a situational crime prevention framework customized for automotive cybersecurity to enable categorization of different strategies based on stakeholders within the possible range of users. These artifacts are combined to inform the development of socio-technical design patterns to work in tandem with the technical solutions. This can include education strategies for high-risk users, focused on identifying factors related to cybersecurity attacks.

The purpose of this work is to provide tools for preventing cybersecurity attacks inside autonomous vehicles. This is done by identifying and enabling a range of stakeholders interacting with these automotive systems. The threat assessment helps identify attacks across the entire spectrum of stakeholders within the systems, ranging from developers to users. The developers can use our automotive cybersecurity design patterns to find common problems and instantiate relevant solution strategies. The solutions can then be categorized or identified using the automotive cybersecurity prevention framework, addressing stakeholders with a technical background and also OEMs, technicians, or employees at a dealership. Then reusable solution strategies that focus on all stakeholder actions provide complementary methods of securing self-driving vehicles for all users, but uniquely address the stakeholders in the broader community. These tools were created by leveraging criminology concepts. For example, applying model-based approaches to criminology-based preventative strategies for cybersecurity attacks on self-driving vehicles provide a multi-pronged approach to address these problems and solutions. These models promote a reusable strategy that is used to further secure these vehicles. With these solutions, we create a range of socio-technical cybersecurity strategies based on the type of interaction for any number of stakeholders. The integration of crime theory provides not only a technical approach but also a human-focused approach, intended to complement and supplement technical strategies previously researched in this field and used in industry.

1.3 Contributions

The goal of our work is to address the challenge of creating reusable strategies to prevent, and in some cases detect automotive security attacks. The contributions of this work are as follows.

1. Assessment of automotive cybersecurity vulnerabilities: An overview of current and future automotive vulnerabilities, cybersecurity attacks, and solutions across a modern vehicle. This includes a threat assessment ranking system to help categorize different characteristics of an attack and/or attacker relating to an attack [14].

2. Automotive cybersecurity design patterns: A set of reusable design strategies for automotive systems to address cybersecurity vulnerabilities of a vehicle, including inward-facing and outward-facing communication [14, 15].

3. Situational crime prevention matrix for automotive cybersecurity crime: A crime theory-based framework to organize and categorize different solutions across a spectrum of stakeholder involvement with various degrees of human interaction to prevent cybersecurity attacks [16].

4. Criminology-Based Domain Model: An instantiation of a Unified Markup Language (UML) domain model [17, 18] that explicitly focuses on the aspects of humans interacting with a given system, specifically focusing on the time and events around a cybercrime occurring.

5. Socio-technical Design Pattern Template: A template to describe reusable stakeholder-based socio-technical design solutions for preventing and detecting cybersecurity threats inside vehicles using crime theory concepts [19].

6. Socio-technical design patterns: A repository of patterns targeting different stakeholders and roles to further enhance the cybersecurity across the lifetime of the autonomous vehicles [19].

1.4 Organization of Dissertation

The remainder of this dissertation is organized as follows. Chapter 2 provides background information on foundational concepts and topics. Chapter 3 describes the threat assessment of cybersecurity vulnerabilities applicable to autonomous vehicles. Chapter 4 presents our automotive cybersecurity design patterns. Chapter 5 describes our situational crime prevention matrix, customized for protecting against automotive cybersecurity attacks. Chapter 6 describes our socio-technical automotive security design patterns. Chapter 7 provides a summary of the work and discusses future work. The appendix has two sets of design patterns: Appendix A gives the complete set of automotive cybersecurity design patterns to date, and Appendix B gives the complete list of socio-technical design patterns for automotive cybersecurity to date.

CHAPTER 2

BACKGROUND AND RELATED WORK

This chapter describes topics that are used as background and foundations for the dissertation research, including design patterns, socio-technical design, and criminology theory concepts.

2.1 Design Patterns

Software design patterns facilitate the reuse of solution strategies for specific problems [20]. By using a known solution to a common problem, developers are able to benefit from successful designs and lessons learned from other developers. The original design patterns by Gamma et al. [20] describe a design pattern with four essential parts: the name of the pattern, the problem addressed by the pattern, the solution strategy, and the consequences of the pattern. Following this approach, several hundred abstract patterns have been implemented for general use in software systems [21]. These patterns are then further analyzed to help investigate future research surrounding design patterns [22]. Then, these patterns are viewed in online and security systems to see how they can be properly and effectively integrated [23]. The fundamentals of design patterns creating reusable problem/solution sets can be extended in detail for autonomous vehicles, and also extended to focus on different stakeholders interacting inside the automotive space.

2.2 Socio-technical Design

The premise of socio-technical systems is that system designs take into account both social and technical factors that influence the functionality and use of the computer-based systems [24]. The technical aspects of these socio-technical systems include hardware and software, while the social dimension involves the cognitive and social aspects of humans when interacting with the system. The field of socio-technical work includes human-computer interaction, developer habits, and stakeholder interactions within a given environment or structure.

2.3 Criminology Terms

This section overviews criminology terms that will be used throughout our work. This includes, the routine activity theory [4], the crime triangle [5], and the situational crime prevention matrix [7]. The criminology terms typically refer to traditional crime, such as burglary, assault, theft, and other non-digital crime.

By leveraging criminal justice theory and terminology, we can incorporate a more human-centric approach to analyzing attacks and risks for autonomous vehicles with respect to cybersecurity. For example, the *Routine Activity Theory* (RAT) analyzes crime based on three key interactions and their relevant routines [4]. The RAT has three core aspects:*the offender*, *the victim*, and *the absence of a guardian* [4]. The crime occurs when these three actors intersect, focusing on their routines and the actions taken prior to the malicious act occurring. When analyzing these routines from the three actors, a clear problem space and responsible parties are identified and allow for the creation of relevant solutions. Figure 2.1 depicts a diagram that captures the relationship between the three elements of the RAT. The focus on routines from specific actors inside a space, as opposed to other factors like demographic information, enables us to extend this work into the automotive space and to a range of stakeholders (e.g., drivers, dealership, third-party vendors).

Furthermore the RAT has been successfully adapted to the cyberspace. For the RAT to be applied, a consistent location is needed for guardians and offenders. While initially the RAT required place, proximity, distance, and temporal order info, it has since been adapted to cyberspace [25]. For example, the RAT has been used to analyze many different types of attacks including phishing, viruses, and fraud when used in tandem with accessible target elements such as those found in VIVA (value, inertia, visibility, and accessibility) [26]. In addition, adapting RAT routines have also helped capture aspects of criminal victimization with internet fraud [27].

Another criminology concept that we use in our work is the *Crime Triangle*, also known as the problem-analysis triangle [5]. The crime triangle addresses how a crime happens and in what setting. This crime triangle can be found in Figure 2.2. The three core concepts of the crime triangle are a *target/victim*, *offender*, and a *place* where the crime occurs. The intersection point is where the actual crime, or event, takes place. Furthermore, each of these aspects can have a managing entity that can make changes to a respective element. For example, an *offender* may have a handler who is able to dictate actions or behavior. The idea of an overarching entity, such as a handler, also applies to the *victim* who should have a guardian, and the *place* should also have a

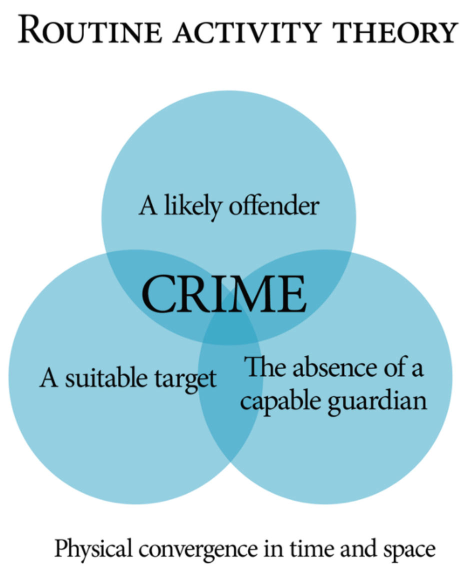

Routine activity theory
A likely offender
CRIME
A suitable target
The absence of a capable guardian
Physical convergence in time and space

Figure 2.1 Routine Activity Theory [4]

manager. Through the combined usage of both the RAT and Crime Triangle, we are able to further define and outline potential threats within the automotive cybersecurity domain. The combination of these two concepts is visually depicted in Figure 2.3. This figure attributes relevant routines to the different aspects of the crime triangle, and also shows where the crime will occur in the respective setting.

Finally, our work leverages and customizes the situational crime prevention matrix originally created by Clarke [7]. The goal of the SCP matrix is to use preventative strategies to stop a crime before it occurs in a given context. The five main categories found in the original SCP Matrix for organizing strategies are **increasing the attacker's effort**, **increasing the risk**, **reducing rewards**, **reduce provocations**, and **removing excuses** [7].

The first category of the proposed strategies revolve around **increasing the effort** of the attack. When a target seems more difficult to obtain, an attacker may be less inclined to commit that crime. This can be done in a number of ways including: *harden targets*, *control access to facilities*, *screen exits*, *deflect offenders*, or *control tools and weapons*. The *harden targets* strategy attempts to make the asset of the victim more difficult to access, for example adding a padlock to a door. The *control access to facilities* ensures that an extra layer or protection is added to the facility access points,

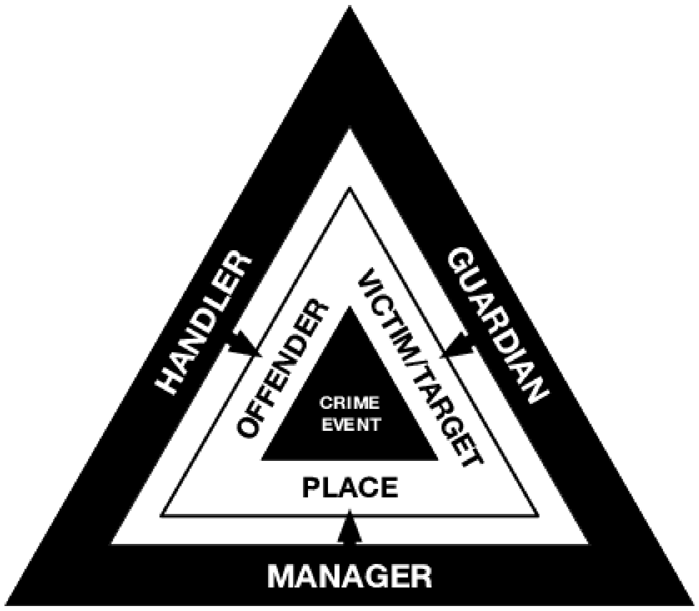


Figure 2.2 Crime Triangle [5]

such as an electronic access card being required. The *screen exits* requires something such as an exit document to leave a given area where a crime could be committed, such as customs required an exit document in an airport. The *deflect offenders* ensures that high risk areas are closed in advance, such as blocking off a street during parades to protect citizens and prevent crashes. Lastly, the category *control tools and weapons* attempts to obtain objects that can be used to commit a crime, such as a restriction on spray paint sales to minors.

The second category, **increase the risk**, refers to the scenario where the criminal is forced to take on on additional risk when attempting to performing an illicit activity. The risk can be increased in any of the following ways: *extend guardianship*, *assist in natural surveillance*, *reduce anonymity*, *utilize place managers*, and *strengthen formal surveillance*. The *extend guardianship* increases the number of people watching a given target, an example could be establishing a neighborhood watch. The *assist natural surveillance* is similar but hopes to establish more visibility in dangerous areas,

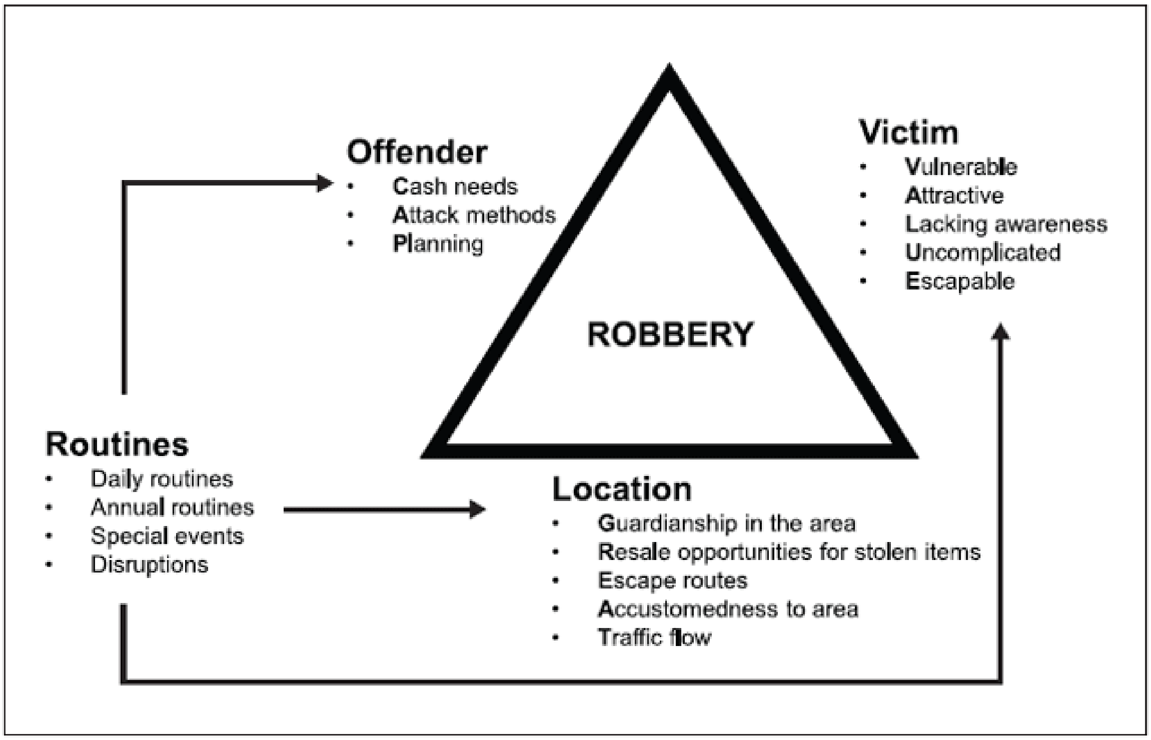


Figure 2.3 RAT to Crime Triangle [6]

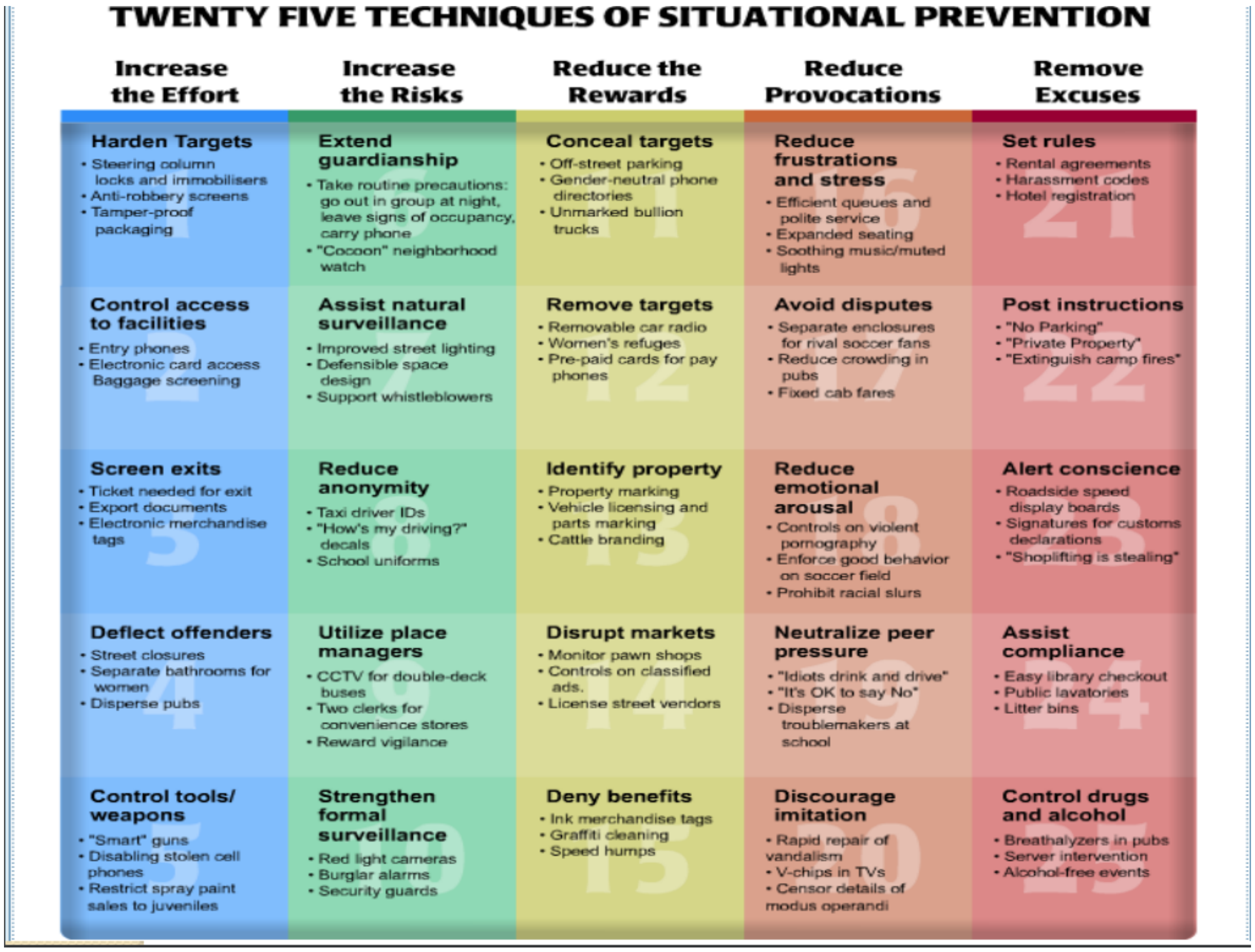
such as fixing or adding additional lights in a given region. Furthermore, by *reducing anonymity*, the risk of committing a crime becomes very high and increases likelihood of being identified prior to the act. Also by *utilizing place managers*, the danger of getting caught increases, for example having additional store clerks in a convenience store. Finally, *strengthening formal surveillance*, such as security cameras or guards makes the crime riskier.

The third category of prevention strategies attempts to **reduce the rewards** a criminal would receive when committing a crime. These are categorized as the following strategies: *conceal targets*, *remove targets*, *identify property*, *disrupt markets*, and *deny benefits*. Through *concealing targets* criminals are not able to quickly assess the value of a given rewards, this can be done in a number of ways including having unmarked trucks to help hide contents. Another strategy includes *removing targets*, such as removing a radio or hiding valuables inside your vehicle. The goal of *disrupting markets* takes away the means of monetary gain for a given object that may have been stolen, this includes pawn shops being monitored for certain high-risk objects. Another strategy is *deny benefits* which includes ink tags on clothes that when removed, ruin the value of an item.

The fourth category of prevention strategies attempt to **reduce provocations** and create an environment that does not allow criminal behavior to begin. This includes the following strategies: *reduce frustrations and stress*, *avoid disputes*, *reduce emotional arousal*, *neutralize peer pressure*, and *discourage imitation*. The goal of *reducing frustrations and stress* as well as *avoiding disputes* attempt to create environments where people are less likely to lash out. This includes having additional bartenders manage lines and quickly serve customers. This could also be extended to *reduce emotional arousal*, which can be done by things like banning racial slurs inside a bar or pub. When *neutralizing peer pressure* the crime is prevented by removing negative actors who may encourage a crime. The *discourage imitation* halts a copycat mentality by either quickly addressing crimes or fixing damages, for example cleaning graffiti off a surface.

The fifth set of strategies focuses on **removing excuses**. Strategies include *setting rules*, *post instructions*, *alert conscience*, *assist compliance*, and *control drugs and alcohol*. By having clear rent agreements and posted signs around a facility specifying legal parking zones encourage both *setting rules* and *posting instructions*. When driving down the road and seeing digital signs showing the speed, this tells a driver to "slow down", *alerting the conscience* of a driver by grabbing attention. Another way to remove excuses includes *assisting in compliance*, for example additional wastebaskets to prevent littering. The final strategy of *controlling drugs and alcohol* can include traffic checkpoints on holidays, alcohol-free events, or cutting off customers at a bar.

These 25 categories create tangible and focused categories for organizing strategies to prevent crimes. The categories themselves help to focus on what relevant details need to be addressed in order to prevent crimes in the future, rather than simply reacting when a crime occurs. A detailed diagram of the SCP matrix can be found in Figure 2.4, which summarizes the categories used and different strategies proposed [7]. Our work leverages these criminology concepts and customizes them for application to the automotive cybersecurity domain. This is then used to develop a framework around stakeholders to create reusable preventative, human-focused defense strategies against automotive cybersecurity crime [16].



TWENTY FIVE TECHNIQUES OF SITUATIONAL PREVENTION

Increase the Effort	Increase the Risks	Reduce the Rewards	Reduce Provocations	Remove Excuses
Harden Targets • Steering column locks and immobilisers • Anti-robbery screens • Tamper-proof packaging	**Extend guardianship** • Take routine precautions: go out in group at night, leave signs of occupancy, carry phone • "Cocoon" neighborhood watch	**Conceal targets** • Off-street parking • Gender-neutral phone directories • Unmarked bullion trucks	**Reduce frustrations and stress** • Efficient queues and polite service • Expanded seating • Soothing music/muted lights	**Set rules** • Rental agreements • Harassment codes • Hotel registration
Control access to facilities • Entry phones • Electronic card access Baggage screening	**Assist natural surveillance** • Improved street lighting • Defensible space design • Support whistleblowers	**Remove targets** • Removable car radio • Women's refuges • Pre-paid cards for pay phones	**Avoid disputes** • Separate enclosures for rival soccer fans • Reduce crowding in pubs • Fixed cab fares	**Post instructions** • "No Parking" • "Private Property" • "Extinguish camp fires"
Screen exits • Ticket needed for exit • Export documents • Electronic merchandise tags	**Reduce anonymity** • Taxi driver IDs • "How's my driving?" decals • School uniforms	**Identify property** • Property marking • Vehicle licensing and parts marking • Cattle branding	**Reduce emotional arousal** • Controls on violent pornography • Enforce good behavior on soccer field • Prohibit racial slurs	**Alert conscience** • Roadside speed display boards • Signatures for customs declarations • "Shoplifting is stealing"
Deflect offenders • Street closures • Separate bathrooms for women • Disperse pubs	**Utilize place managers** • CCTV for double-deck buses • Two clerks for convenience stores • Reward vigilance	**Disrupt markets** • Monitor pawn shops • Controls on classified ads. • License street vendors	**Neutralize peer pressure** • "Idiots drink and drive" • "It's OK to say No" • Disperse troublemakers at school	**Assist compliance** • Easy library checkout • Public lavatories • Litter bins
Control tools/ weapons • "Smart" guns • Disabling stolen cell phones • Restrict spray paint sales to juveniles	**Strengthen formal surveillance** • Red light cameras • Burglar alarms • Security guards	**Deny benefits** • Ink merchandise tags • Graffiti cleaning • Speed humps	**Discourage imitation** • Rapid repair of vandalism • V-chips in TVs • Censor details of modus operandi	**Control drugs and alcohol** • Breathalyzers in pubs • Server intervention • Alcohol-free events

Figure 2.4 Situational Crime Prevention Matrix [7]

CHAPTER 3

AUTOMOTIVE THREAT ASSESSMENT

This chapter provides an assessment of automotive cybersecurity vulnerabilities. A number of assessment criteria are introduced, vulnerabilities are identified, and potential impact of breaches are reviewed. A simple overview of different threats can be helpful, but fails to address the traits around the attacker who is needed to actually exploit this vulnerability. The traditional threat assessment focuses on components that can be attacked or exploited to help find flaws, but our approach additionally focuses on the human dimension and those factors around the cybersecurity vulnerability within the automotive vehicle. By broadening the focus from the type of attack to the threat vector itself, we can better assess the likelihood of an attack around the offender to better encapsulate possible danger. Furthermore, through our collaboration with Ford Motor Company [28] we were able to obtain input on the practicality and applicability of our assessment criteria. With this threat vector-focused assessment of attacks, vulnerabilities can be identified inside and outside the vehicle, and preventative approaches can be developed, specifically by extending beyond technical solutions.

3.1 Classification and Criteria

To estimate the severity of the vulnerabilities in vehicle security found to date, we identify a set of criteria to assess a given vulnerability. In order to illustrate the assessment utility, we overview two potential attacks on an automotive system. One attack involves the telematics unit to gain control of a vehicle, and the other uses key fob signals to gain entry inside a vehicle [11]. Then these assessment criteria are applied to the attacks reported to date to obtain a summary of the threat that provides a high-level perspective of criticality.

1. Knowledge - Knowledge required refers to the technical expertise required to execute an attack. Compromising the telematics unit requires duplicating modem links and protocols [11] , that entails a considerable amount of effort. Duplicating a key fob signal, however, is an easier task than cracking the telematics unit as certain cracking devices are already available in the open market [29]. In this example, the knowledge a criminal needs to gather will be much less if he unlocks the car door using a duplicated key-fob signal compared to compromising the entire telematics unit.

2. Access to vehicle required - This defines the modicum of physical, indirect, or remote access needed to gain entry to some entity to perform a successful attack. The telematics unit can be cracked from a global range by placing multiple calls. The key-fob duplication requires the attacker to be present physically within a close range of an intended target vehicle.

3. Time Required for Attack - Time required to complete an attack from the moment it is set in motion is generally different for each method. In cracking the telematics unit, multiple calls need to be placed to the vehicle. The key-fob duplication attack time depends on how fast the cracking algorithm can gain access to the right radio frequency.

4. Safety Risk / Severity of Threat - The safety and severity of threat refer to the degree of impact on the safe operation of a vehicle in case of a cyber attack/threat. An attack that leads to an adversary gaining control of critical functionality like braking, steering, etc. are considered to be more critical to safety than others that may create only nuisance to driving like changing of dashboard display. In the example, the compromise of the telematics unit is more critical since the functions can be altered even when the vehicle is in motion, in addition to other adverse effects. When looking at the key-fob duplication attack, this can only occur when the vehicle is stationary, with engines off. This is less critical to the safety of the driver.

5. Type of Attack - This defines the four types of attacks that can occur within cyber-based systems. These include interception, interruption, modification, and fabrication [30]. The term interception refers to the hijacking of data being sent from one destination to another. This can include packet sniffing or eavesdropping on other communications. An interruption attack includes stopping an element of the system from functioning properly, either through a network or similar means. For example, overloading a server host to cause a crash or cutting a communication line would be examples of an interruption attack. The modification attack includes altering the functionality of a system such as changing an incoming message, while fabrication creates something new such as creating user credentials to gain access to a system.

3.2 Threat Surfaces

The threat surfaces for a cyber attack on a vehicle are those interfaces or components of the vehicle that can be used by an attacker to launch a cyber attack against it. Essentially, they are the gateways or means through which a cyber attack gains entry into the underlying electronics or software code embedded in various components of the vehicle. In this section, we describe the attack surfaces identified thus far, both in experimental or research settings. To break any aforementioned surface or use it as a channel for an attack, any potential attacker requires access to the surface for reverse-engineering or other manipulation purposes. The threat surfaces are categorized by level of access needed to the vehicle as described above.

3.2.1 Direct Access to Vehicle

To execute certain attacks successfully, a hacker will need to gain physical access to the vehicle and the unit or module through which he intends to route the attack through. Then they can go ahead and physically modify or plug-in certain devices into that surface. This can be done by that person or done via a proxy. These direct access surfaces are identified as the units or modules vulnerable to these direct access attacks, the vectors are as follows:

a. OBD-2 Port : The On-Board Diagnostic (OBD-2) is a diagnostic port that gives the vehicle owner or support technicians access to the status of various vehicle subsystems. Generally, it is located under the steering column or under the dashboard at the passenger side. However, it can be located anywhere in the vehicle depending on ease of access and these ports have been made mandatory for all vehicles manufactured or sold in the United States after 1996 [31]. Many cyber attacks have been demonstrated to be possible by plugging in a laptop or other handheld device into this port. These demonstrations were done in experimental settings as shown in [11, 32, 33]. It is to be noted that constant physical access to the OBD-2 port is no longer necessary to execute a successful attack. The availability of Wireless Fidelity (WiFi) and Bluetooth enabled OBD-2 attachments [34] makes it possible to execute attacks remotely from within wireless range once the attacker has plugged in the device into the target vehicle's OBD-2 port.

b. CAN-Bus : The CAN-bus is the communication backbone between and within vehicle sensors, actuators and Electronic Control Units (ECU) [35]. It relies on a broadcasting protocols that allows for any ECU attached to the CAN Bus to both send and receives messages as predefined ECU behavior, independent of the CAN Bus [36]. The CAN Bus is designed to be lightweight and fast, and consequently lacks common communication protocol features such as authentication or encryption. While data on a CAN Bus may carry identifying information to be interpreted by the intended receiving ECU, the CAN itself does not identify particular data frames, instead relying on a prioritization scheme of message arbitration to control access to the CAN bus [37]. The CAN-Bus is implemented using wires embedded in the vehicle body. The potential security drawbacks to such a communication model include vulnerabilities posed by the lack of message authentication, such as spoofing and injection attacks. Furthermore, communication within the CAN-bus can be eavesdropped or a foreign node can be planted on the bus to carry out malicious activities [33]. The planting of a node or intercepting the CAN-bus wiring requires at least one-time physical access to the vehicle by the hacker. The cascading damage caused by access to the CAN-Bus can range anywhere from stealing driver data, modifying ECU data, and even reprogramming the software of various ECU's [38]. For example, an injection attack makes it possible for an adversary to manipulate the functionality of an ECU [39]. The CAN message arbitration system can also make it vulnerable to denial of service (DoS) attacks through an attacked ECU, such as the tire pressure monitoring system (TPMS) [38, 40].

c. Media player/ Auxiliary port : Most vehicles have an on-board system for playing digital music CDs and many have an auxiliary port to plug in digital music players or mobile phones. These systems are attached to the CAN-bus and have been demonstrated to be vulnerable to cyber attacks [11, 41, 42]. The media player can be exploited by playing compromised digital audio files. Direct access to the digital media player in the car is necessary to play back digital audio containing the malicious code. However, when CD is played, the code is not legible to human ears [11]. This attack can also be executed without physical access as it requires only an engineered compact disc with a compromised track to be played back. Therefore, a hacker can resort to social engineering to ensure that the compromised disc is played by the driver or owner. The functionality of the media player also allows attackers to gain access to critical system functions through only the media player. The varying degrees of severity from an attack originating in the media player make this surface a critical element to be secured.

3.2.2 Indirect Access to Vehicle

Indirect access for a cyber attack on a vehicle is defined here as attack procedures where physical access to the vehicle is not necessary. The requirement is that the attacker can infect, or have access to certain devices, that are within the range of the vehicle. This can also include anything that is connected to the vehicle that can then, in turn, be used to attack the internals the car.

a. Internal Bluetooth Network: Most modern vehicles have an internal Bluetooth network to support devices like cellular phones. These devices can integrate with functions that are provided with the car such as hands-free speaking, among several other utilities of Bluetooth network in a vehicle [43]. The Bluetooth network has been cracked in experimental setting [11], but the process needs indirect access to the vehicle. A hacker needs to compromise a device of the owner or driver that the latter would use once they are inside the vehicle to gain access to the vehicle Bluetooth network. Once this device has been paired, the hacker can manipulate or brute force the security pin of the network to gain full access to it.

b. Pass-thru device at dealer location : A pass-through device is used in conjunction with a computer to reprogram vehicle control modules through the OBD-2 port. It acts as an interface between a service computer used by a technician and the vehicle ECUs. Currently, technicians extensively use these pass-thru devices. Most pass-thru devices are vendor-specific with some standardization. A car can be attacked by a hacker using a servicing window, the time interval during which a car is connected to dealer's service computers and when a car is maintained at a dealer location. In this indirect method of attack, a service computer can be compromised or infected by malicious code that will execute when the computer is in the process of diagnosing or flashing some under-service vehicle's ECU [11].

3.2.3 Remote Access to Vehicle

The other cyber attacks can be executed without any physical access, direct or indirect, to the intended target. These are remotely executed attacks where the range from which the attacks can be executed vary from a short distance of about 20m to a global range where internet is available.

a. Key-fob : A significant number of vehicles use the low cost Radio Frequency Identification (RFID) based keys. These keys are used either in lieu or in conjunction with normal physical keys for vehicle door lock or unlock and as engine immobilizers. The key device, also known as the key-fob, contains an embedded RFID chip. A large majority of the car vendors use either the Keeloq or Digital Signal Transponder (DST) technique for their car locking system [12, 44, 45]. The Keeloq algorithm has already been cracked by researchers and the knowledge is available in the public domain [44, 45]. Hackers can use any of these methods to break this algorithm [45]. Also, DST has been shown to be compromised either by reverse engineering or by key-cracking [45]. These exploits can be carried out remotely from an appropriate range [11, 44].

b. Tire Pressure Monitoring System (TPMS) : The tire pressure monitoring system is a feature available in most modern vehicles that monitors the pressure in the tires. The sensors monitoring the tire pressure transmit monitoring data wirelessly to receivers in the dashboard area. Depending upon pressure data, either a warning light is displayed in the dashboard or the data itself is shown. The communication between TPMS sensor and receiver has been shown to be vulnerable to interception and spoofing as well as the functionality of TPMS being rendered defunct. Furthermore, displaying erroneous or spurious warnings and triggering CAN packet transmissions from other modules are other possible exploits of the TPMS security issue [11, 46].

c. Telematics System : The telematics system in a car generally encompasses services like in-vehicle security, hands-free calling, turn-by-turn navigation, and remote diagnostics. These services are generally subscription based and are provided via a wireless carrier. A few well-known telematics service brand names provided by leading automotive manufacturers include OnStar, Uconnect, SYNC, and Bluelink [47]. The telematics system has been shown to be a viable attack surface in multiple experimental setups. Checkoway et al [11] was able to upload malicious code via multiple calls placed through the telematics unit remotely. However, the successful attack requires multiple calls to the vehicle and is dependent on default timeout for calls as setup in the unit. In another example a vehicle was located via the internet using its IP address and then the telematics unit was exploited to remotely control various functions of the car including braking and acceleration [48, 49, 50].

d. FM Radio / Digital Radio Receiver: The use of radios in vehicles are ubiquitous and most vehicles contain a digital radio receiver. This component is another target for cyber attack and was shown by using a simple malicious digital broadcast which was received by the target vehicle and was able to compromise its electronics [50, 51, 52]. This attack is vehicle model specific, however did show the vulnerability of this component.

e. Vehicular Ad-Hoc Networks: Vehicle-to-vehicle (V2V) and Vehicle-to-infrastructure (V2I) communications can be fulfilled quickly by establishing temporary networks known as vehicular ad-hoc networks. Vehicle Ad-hoc Networks (VANETs) make use of a number of wireless inter-vehicle networking, such as DSRC (dedicated short range communications (DSRC), long term evolutionary vehicle (LTE-V), and WiMAX (Worldwide Interoperability for Microwave Access) [53]. With the increasing movement towards vehicle-to-vehicle (V2V), vehicle-to-infrastructure (V2I), and vehicle to third party devices (V2X) communication, [54, 55] security and privacy challenges correspondingly increase [53, 56]. These networks, like other computing domains, are vulnerable to cyber attacks [44]. The problems arise in securing downloadable apps and updates from these vendors remotely, which can download and make them susceptible to being used as Trojans or to deliver viruses [57] .

f. Over-the-Air Firmware Updates: Several new vehicle models have the ability to update their ECU firmware wirelessly. The updates are transmitted from the manufacturer from a central server. This allows for interception, modification, or duplication of these updates by hackers as these changes are being pushed to vehicles [58].

3.3 Threat Assessments

To estimate the severity of the threats in vehicle security found to date, we define a rough digital representation of the classification factors (access, knowledge, etc.) discussed above. Then these classifications are applied to the attacks reported to obtain an overview of critically dangerous threat surfaces. Table 3.1 gives a detailed explanation for the system of rating threats based on access, knowledge, time, and severity. Next, Table 3.2 shows an overview of the different onboard components, types of threats, and possible outcomes of successful attacks. Finally, Table 3.3 gives a sample assessment showing the given rating system and the vulnerability of each component based on attacks found in the literature. This assessment shows how different threats can be individually categorized within our system.

Table 3.1 Threat Assessment: Access Overview

Access	**Knowledge**	**Time**	**Safety Risk**
1. Constant physical access to the vehicle is required. EG: Hacker needs to be present in the vehicle or a device belonging to the hacker needs to be present inside the vehicle	1. Not performed before and possibly required low level understanding of architecture and/or proprietary code/modules	1. High execution time including multiple retries aimed at brute forcing/guessing entry code. Successful attack also depends on other factors which are dependent on chance.	1. Minor issues like improper display and annoyance. Excluding critical displays like speed and rpm.
2. One time physical access is required followed by constant proximity of 100 meters. EG: Planting a device on the body of the car and then following the car while attack is being executed.	2. Attack is known to have been performed but no other details available. Requires low-level understanding of architecture and/or proprietary code/modules.	2. Execution of attack depends purely on chance.	2. Minor issue but may mislead the driver into taking an action.
3. Only close proximity is required and no physical access to the vehicle. EG: Only close proximity is needed to the vehicle in order to execute the attack, like compromising the TPMS.	3. Attack is known to have been performed before and basic details of the attack surface targeted, procedure/tools used, and consequent results is available. A low-level understanding of modules/code is required. * 3. High execution time including multiple retries aimed at brute forcing/guessing entry code. No other factors involved.	3. Issues that may impair driving abilities of the driver to some degree or slightly impairing vehicle functionality such that vehicle responses to every maneuver but not optimally.	Issues that may impair a drivers abilities to some degree or slightly impair vehicle functionality such that vehicle responds to every maneuver but not optimally.
4 . Only time physical access is required and then can be access from world wide web. EG: Either planting a device on the body of the vehicle or obtaining some information from inside the vehicle and then being able to exploit that attack from a long range.	4. Attack is known to have been performed before and detailed analysis of the attack surface, procedure/tools used, and consequent result is available. A low-level understanding of codes and modules is required to perform any small changes.	4. Medium execution time as it does not depend on multiple retries and exact time required can be determined beforehand.	4. Issues that may lead to severe impairment of non-critical vehicle feature of functionality. Also includes compromise of private data.
5. Can be accessed via the world wide web. No other complimentary device/entity is required anywhere near the vehicle. EG: The kind of access is actually no access to the vehicle. The attack can be pinpointed and executed from anywhere.	5. Attack is known to have been performed before and semantics of attack and method of piercing of attack surface is known. Basic minimal knowledge is required (if any) of underlying code/architecture.	5. Very short execution time and is synchronous/almost synchronous in effect.	5. Issues that can severely compromise critical functions of the vehicle like braking, acceleration, etc.
	6. Attack is known to have been performed before and tool(s) manuals required for attack is available in public domain. Any knowledge required can be acquired from manual in a very short time.		

Table 3.2 Threat Assessment: Threat Overview

Component	**Threat Surface**	**Types of Threats**	**Effects**
OBD-2 Port	Direct Access/WiFi Access (via Pass-Thru Device).	Interception: Bus Data including GPS Data, Interruption: DoS Attack can be implemented, Modification: Flashing of ECU's, Fabrication: Reprogramming of ECU's [11] [32] [33]	OBD-2 is the diagnostic gateway of the whole embedded electronic system of the vehicle. Hence multiple malicious effects are possible. [11] [32] [33] [34]
Key-Fob	Duplicate RF-ID Chips (mainly the algorithm)	Interception: tools available to catch signal from legitimate key fobs, Fabrication: Duplicate RF-ID chips used to unlock cars [12, 44, 45]	Theft, Internal access which can be combined with another attack. Also can gain access to electronic components. [12, 44, 45]
Media Player/AUX Port	Compromised media connected to media player/AUX port.	Interruption: Entertainment system can be disrupted, Fabrication: software can be modified for cyber-attacks [41] [42] [38]	Component service can be interrupted or a disturbance can cause attacks on ECU's (either directly or indirectly. [41] [42] [38]
Pass thru device at dealer location	Connected to a compromised computer or local computer/device.	Interruption: computer attaches to pass-thru and effects function, Modification: malicious changes to ECU's or flashing [11]	Access to car diagnostic port. [11]
Telematics Unit	Compromised cell phone or malicious Bluetooth device that has gained access to car's internal Bluetooth network.	Interception: calls and data can be intercepted, Interruption: calls and data can be interrupted, Modification: Software components can be flashed or compromised [29] [47] [43]	Disturbing telematics functionality, can gain complex access to all other ECU's of vehicle. [29] [49] [43]
Internal Bluetooth Network	Any malicious device in close proximity could be brute forced to gain access (using the PIN)	Interception: Bluetooth traffic can be intercepted, Interruption: Network can be attacked with DoS and fuzzing attempts [43]	Immediate compromise of telematics and hands free, also access to other electronic components. [43]
ECU's	Duplicate component installation/ access via some attack surface	Modification: change for unintended purpose, Interruption: disable or flash, Fabrication: malicious code can be implanted [39] [38]	complete breakdown of ECU related services. [39] [38]
TPMS	Direct radio access to its transmission to the vehicle dashboard monitor	Interruption: TPMS data flow can be stopped, Fabrication: Wrong data can be transmitted, Interception: data can be spoofed [11] [46]	ECU can be rendered useless and inaccurate with false messages being displaced. [11] [46]
Vehicular Ad-Hoc Networks	Compromised transmission via network node	Interception: possible interception of data, Interruption: disturbed between nodes, Fabrication: Wrong data can be transmitted by malicious entities [53] [55] [54]	Location tracking and bogus information feeding, Proliferation of worms or virus. [53] [55] [54]
Telematics Services	Compromised via IP address of vehicle over the internet. Android app has been used from mobile provided by service.	Interruption - Using telematics services can be interrupted [11] [49] [50]	Since access to every BUS has been demonstrated, braking, acceleration, etc. can be effected. [48] [49] [50]
Radio	Compromised via broadcasting of malicious data	Fabrication: False radio signals from audio can be broadcasted. Interception: vehicle functioning can be interrupted using the radio signals [50] [51] [52]	Steering and braking the two safety critical components has been shown to be compromised. [50] [51] [52]

Table 3.3 Threat Assessment: Attack Breakdown

Surface	**Access**	**Threat Type**	**Knowledge**	**Access**	**Time**	**Safety Risk**	**Overall Risk**	**Author**
Bluetooth	Close-Range	Interception	3	3	4	2	12	Dardanelli [59]
Bluetooth	Close-Range	Modification	3	3	4	5	15	Dardanelli [59]
Key-Fob	Physical	Fabrication	3	1	3	2	9	Francillon [60]
Key-Fob	Remote	Fabrication	2	5	3	2	12	Francillon [60]
Telematics	Remote	Modification	2	5	2	5	14	Greenberg [48]
Key Fob	Close-Range	Fabrication	2	5	4	5	16	Greenberg [48]
OBD-2 Port	Physical	Fabrication	2	5	2	5	14	BBC News [61]
Radio	Remote	Fabrication	2	5	2	5	14	Vallance [50]
Key-Fob	Close Range	Fabrication	4	3	4	2	13	FireEye [62]
Telematics	Remote	Interception	2	5	5	1	13	FireEye [62]
Telematics	Remote	Modification	2	5	2	5	14	FireEye [62]
ECU	Close-Range	Modification	1	3	2	4	10	FireEye [62]
User	Remote	Fabrication	5	5	5	3	18	FireEye[62]
Infotainment	Remote	Interception	2	5	4	4	15	Armstrong [63]
VANET	Remote	Interruption	2	5	4	4	15	Hasbullah [64]
OBD-2 Port	Physical	Fabrication	2	5	4	5	16	Leyden [52]
ECU	Close-Range	Modification	2	2	3	5	12	Leyden [52]
Telematics	Physical	Modification	2	2	3	5	12	Leyden [52]
Telematics	Remote	Modification	2	5	3	5	15	Leyden [52]
Radio	Remote	Fabrication	2	5	4	5	16	Leyden [52]
OBD-2 Port	Physical	Modification	2	1	3	2	8	Klinedisnt [65]
OBD-2 Port	Close-Range	Interruption	2	3	3	4	12	Klinedinst [65]
OBD-2 Port	Remote	Fabrication	2	5	3	5	15	Klinedinst [65]
GPS	Remote	Modification	1	5	3	4	14	UT Media [66]
Sensors	Close-Range	Interruption	5	3	5	5	18	Lambert [67]
CAN-Bus	Close-Range	Interruption	4	3	4	1	12	Larson [68]
CAN-Bus	Close-Range	Interception	3	3	3	4	13	Larson [68]
CAN-Bus	Close-Range	Fabrication	2	3	2	4	11	FireEye [62]
CAN-Bus	Close-Range	Modification	3	3	3	1	10	FireEye [62]
ECU	Close-Range	Interception	4	3	4	2	13	Hoppe [69]
OBD-2 Port	Physical	Modification	4	1	4	5	14	Zhang [70]

CHAPTER 4

AUTOMOTIVE CYBERSECURITY DESIGN PATTERNS

As autonomous systems and features become further integrated into vehicles, software and electronics will account for over 50% of the design overhead [59]. However, the current software and its respective sub-systems have been demonstrated to be vulnerable to cyber threats in controlled testing environments [38, 60, 61]. These types of attacks demonstrate a clear increase in threat vectors and attack surfaces for autonomous features and systems, while illustrating the potential for impacting human safety [62]. In order to support systematic and reusable development practices focused on automotive cybersecurity needs, this chapter introduces automotive-focused security design patterns to address cybersecurity vulnerabilities associated with inward facing (i.e., intra-vehicle) and outward facing (i.e., inter-vehicle) communication.

The increasing automotive security vulnerabilities motivate the development of prevention, detection, and mitigation techniques that take into consideration automotive-specific constraints. Formalizing and abstracting common problem and solution strategies (i.e., designs) into design patterns facilitates rigorous development practices and promotes design reuse [63]. Common software security patterns enable developers to rigorously harden a system against security vulnerabilities [64]. While these security patterns, along with security solutions such as encryption and secure network protocols, have helped to harden many software systems, the unique architecture of automotive systems and constraints on system performance necessitate application-specific design strategies [65].

In order to leverage successful security patterns, while also addressing the automotive constraints, we have specialized existing security patterns to account for automotive systems' unique architecture and constraints. This framework for hardening security design and assessing security features can be applied to current automotive systems comprising numerous ADAS (automated driver-assistance systems) elements and extended to the emerging fully-autonomous vehicles. Automotive-specific security patterns enable developers to explicitly and rigorously address security as part of the development process for automotive systems in a proactive fashion. We introduce a template to describe the patterns; the template extends a previously-developed security patterns template [21, 66] to include fields, classification schemes, and information specific to the automotive domain [2]. In addition to the textual descriptions, we also include UML (Unified Modeling Language) [67] class and sequence diagrams to respectively describe the structure and the behavior of a given solution strategy.

The automotive cybersecurity patterns leverage and extend previously-developed security patterns to address vulnerabilities specific to automotive systems, with solution strategies that adhere to constraints and contexts relevant to automotive systems and relevant operating environments. As a preliminary step, we performed an extensive review of the research literature (i.e., state of the art) [68, 69], technical reports from regulation and standards bodies (e.g., [2, 70, 71]), and obtained input from our industrial collaborators (i.e., state of the practice). From these sources, we identified threat vectors and attack surfaces, as well as techniques and/or solution strategies to address the respective vulnerabilities. We distinguish vulnerabilities and solutions that apply to onboard (inward) facing communication (e.g., wheel speed sensor and throttle position sensor signals sent to power train control module) versus outward facing communication subsystems (e.g., telematics units and navigation systems that communicate with cell towers and global positioning systems, respectively). Based on the state of the art and state of the practice reviews, as well as existing security patterns, [21, 64, 66] we collated the information to define a set of automotive cybersecurity patterns. In order to facilitate their use, we provide a number of descriptors for each pattern that are commonly used in security development, including threat types using Microsoft's STRIDE framework[72], Viega and McGraw security principles [73] promoted by a given pattern, and automotive cybersecurity development guidelines as described in SAE J3061 [2].

The chapter presents an ongoing effort with international industrial collaborators working in automotive cybersecurity, including both Tier 1 suppliers and OEMs (original equipment manufacturers) to develop a repository of security patterns.[1] These patterns are being incorporated into development practices that focus on security and its impact on safety for autonomous vehicles,

[1]Our project sponsors involve international researchers and developers, as well as customers, who have provided feedback regarding the patterns.

involving different levels of autonomy.

4.1 Security Patterns

While the design patterns developed by Gamma et al [20] are intentionally general with the goal of being applicable to a wide range of problems, domain-independent constraints need to be considered for our design. In the case of security patterns, threats to a system need to be monitored using specific security mechanisms for the specific context [64]. Following a similar format for describing a design pattern, security patterns are also given a descriptive name, the security problem addressed (e.g., a class of specific threats), a solution strategy to prevent, detect, and/or mitigate the security vulnerability, and a set of consequences [64].

Research in the field of security patterns has been active for almost two decades [13, 22, 74]. The focus has been on capturing higher level security mechanisms and organizing them into abstract patterns, which has guided cross-application security patterns [75, 76]. The aforementioned work underscores the wide use of security patterns throughout different software systems, including distributed systems [77], enterprise systems [23, 64], and more recently, cloud computing systems [21, 64]. Security patterns have been developed for all phases of the software development process, such as requirements gathering, design, and implementation [13]. Ito [22] surveyed trends in security pattern research over the past decade.

Similar to general design patterns [20], security patterns have been described using specific fields to capture use-cases, consequences of design, etc. We leverage and extend a specific template developed by Wassermann and Cheng [66] and then refined by Konrad et al. [78] that captures key areas of a security pattern.

4.2 Automotive Security Pattern Template

The automotive security patterns are described in a template similar to that used by Konrad et al. [78]. The template is used to describe the automotive security patterns, the security principles, and automotive security development guidelines used to characterize the patterns. In addition to the template description, we overview the Viega and McGraw security principles [73], the STRIDE threat categories [72], and the SAE J3061 development guidelines for automotive security [2], all of which is incorporated into the patterns. The revisions and alterations to this template are to help create the most current and effective security procedures. The basic template structure for the following automotive security design patterns is as follows:

- **Pattern Name** - The name and classification of a pattern as either behavioral or structural.
- **Intent** - How the pattern can be used and what security problem it addresses.
- **Motivation** - Background on the security problem, and a mechanism to solve, including examples and security principles that are related.
- **Properties** - STRIDE properties [72] that are addressed through this pattern.
- **Applicability** - Indicates the type of solution proposed: prevention, detection, and/or mitigation of a given security vulnerability.
- **Structure** - Includes the UML class diagram for the pattern and the related descriptions of participants and collaborations.
- **Behavior** - Includes a UML sequence diagram and descriptions of an illustrative scenario with the pattern.
- **Constraints** - Denotes constraints placed on the pattern implementation stemming from the limited resources and real-time constraints of the automotive system.
- **Consequences** - Discusses the effects of the pattern on the areas of Accountability, Confidentiality, Integrity, Availability, Performance, Cost, Manageability, and Usability following the template from Wasserman and Cheng [66].
- **Known Uses** - Discusses examples of this pattern's use in an automotive setting.
- **Guiding Principles** - The guiding principles related to the given security pattern from from Viega and McGraw [73] and also the respective J3061 property [2].
- **Related Patterns** - Any design patterns or security patterns that relate to the given pattern.

Table 4.1 STRIDE Threats and Properties

Threat	**P**roperty	**E**xample Security Question(s) Addressed
Spoofing	Authentication	Does the system use multi-factor authentication? Does the system enforce secure credential creation, use and maintenance principles?
Tampering	Integrity	Can the system detect and prevent parameter manipulation? Does the system protect against tampering and reverse engineering? Were secure software design principles followed during development, including third-party software?
Repudiation	Non-Repudiation	Does the system verify and log all user actions with attribution?
Information Disclosure	Confidentiality	Does the system follow standard encryption practices to secure connections?
Denial of Service	Availability	Was the system built and tested for high availability (e.g., fuzz testing and load testing)?
Elevation of Privilege	Authorization	Does the system support the management of all users and privileges?

4.2.1 STRIDE Threat Framework

The integration of the STRIDE principles help us tailor these patterns to automotive cybersecurity [72]. The STRIDE concept is a component of Microsoft's Security Development Lifecycle [72] framework used to categorize security threats in a system. Along with threat categorization, STRIDE incorporates a set of information technology properties that correspond to the given threats. Table 4.1 outlines the STRIDE threats, the corresponding information technology properties, and example security questions related to the properties. While STRIDE is applicable to general cybersecurity vulnerabilities, other efforts have also explored how STRIDE can be applied to the automotive domain, such as the SAHARA project that extends automotive hazard and risk analysis with STRIDE-based analysis [79].

Using the STRIDE framework in the pattern template assists developers in connecting the threats and their corresponding properties to patterns. Our industrial collaborators suggested the use of the STRIDE system in the pattern descriptions given their common use in industrial contexts. The STRIDE descriptors further facilitate the identification of appropriate patterns to use for a given problem/solution context.

4.2.2 Guiding Principles

This section is similar to that used by Konrad [66, 78], following several security principles used to help motivate the problems that the security patterns are intending to solve. The following security principles from Viega and McGraw [73] are incorporated into the template:

- **V1 - Secure the Weakest Link** - Securing the most vulnerable piece of an application(e.g. network access points or interfaces).
- **V2 - Practice Defense in Depth** - Promote several layers of security throughout an application, which will ensure redundant security if one layer fails.
- **V3 - Fail Securely** - In the case of a system failure, ensure that unauthorized access to the system or information is not possible.
- **V4 - Follow the Principle of Least Privilege** - Give actors in a system the lowest degree of access/clearance necessary to carry out a given task.
- **V5 - Compartmentalize** - Keep sub-systems separate from each other so that a security failure in one will not compromise the others.
- **V6 - Keep it Simple** - Using unnecessarily complicated procedures or systems may lead to unforeseen security vulnerabilities.
- **V7 - Promote Privacy** - Ensure personal information and sensitive data is only accessible by those with ownership or given consent.
- **V8 - Remember that Hiding Secrets is Hard** - Security critical information such as passwords are not easily kept safe and secret. Therefore, designers must be wary of compromised accounts/subsystems affecting the whole system.
- **V9 - Be Reluctant to Trust** - Sub-systems should not rely on other sub-systems being secure and must enforce security against all actors that interface with them.

- **V10 - Use Community Resources** - Community resources are often more secure than individual solutions due to the large number of developers.

We also used guiding principles for developing secure automotive systems defined in SAE Standard J3061[2] to explicitly relate the security patterns to the industry standards for automotive systems. Our industrial collaborators also indicated that applying J3061 would potentially facilitate more industrial collaborations between development organizations. While the principles share common principles/objectives with those from Viega and McGraw [73], J3061 captures the principles in the specific context of the automotive system. The J3061 principles [2] are as follows in Table 4.2:

- **J1 - Protect Personally Identifiable Information and Sensitive Data** - Ensure personal information and sensitive data is only accessible by those with ownership or given consent.
- **J2 - Use Principle of Least Privilege** - Give actors in a system the lowest degree of access/clearance unless necessary.
- **J3 - Apply Defense in Depth** - Promote several layers of security throughout an application. This will ensure redundant security if one layer fails.
- **J4 - Prohibit Software Changes that have not been Thoroughly Analyzed and Tested** - Preventing software from being changed without being thoroughly tested prevents unforeseen security flaws in a system.
- **J5 - Prevent Vehicle Owners from Making Unauthorized Changes** - Along with J4, changes to a vehicle that are not planned thoroughly may miss critical security flaws leading to system vulnerabilities.

4.3 Overview of Automotive Security Patterns Repository

The current repository of automotive security patterns has been drawn largely from automotive security solutions described in recent research literature [64, 77]. These patterns have been developed with guidance from our industrial collaborators, who have developed additional patterns for internal use, as well as augmented their development process to include the use of our security patterns. Table 4.2 categorizes the automotive security patterns to date in the context of their applicability (i.e., prevention (Prev), detection (Det), and/or mitigation (Mit) of security vulnerabilities); related security principles/guidelines as defined by Viega and McGraw (prefixed with 'V'); by the SAE J3061 automotive cybersecurity guidelines (prefixed by 'J'). Next, we include a brief description for each of the patterns in the repository, including citations to known realizations in the automotive domain.

- Authorization provides a structure that facilitates access control to resources [1].
- A Blacklist pattern intends to keep track of the traffic of potentially malicious addresses in a network. Nodes in the network use the *Blacklist* to block traffic originating from the malicious nodes [80, 81].
- The DDoS Redundancy pattern is intended to make a resource or network more resilient to a Distributed Denial of Service attack (DDoS) by providing redundant resources in case a resource becomes inundated with service requests[82, 83]. Pattern Firewall designs a structure that allows for network traffic to be filtered by a set of predefined rules to prevent malicious intrusion (e.g., securing ECUs individually[84].)
- The Multi-Factor Authentication pattern is used to provide a redundant security measure in authenticating an actor or a message. By enforcing an additional level of authentication, malicious actors can be prevented from attacking if a node or single credential is compromised [85].
- Multilevel Security is intended to provide a mechanism for handling access in a system with various security classification levels[86].
- Signature IDS provides a mechanism for detecting anomalies in network traffic by using a base-line characteristic of the (communication) traffic[87, 88].

Table 4.2 Applicability and Viega McGraw [1] / J3061 Principles of Select Patterns[2]

Pattern Name	Applicability	V1	V2, J3	V3	V4, J2	V5	V6	V7, J1	V8	V9	V10	J4	J5
Authorization	Prev				X	X		X					
Blacklist	Mit/Prev		X			X				X			
DDoS Redundancy	Prev/Mit		X	X		X							
Firewall	Prev/Det	X			X					X			
Multi-Factor Authentication	Prev		X			X				X			
Multilevel Security	Prev/Mit				X	X		X	X	X			
Signature IDS	Prev/Mit/Det									X			
Symmetric Encryption	Prev							X		X			
Tamper Resistance	Prev/Det/Mit			X	X							X	X
Third Party Validation	Det/Mit							X		X			

- Symmetric Encryption is used to encrypt messages (between vehicles and with infrastructure) so that only a sender and receiver can read the contents[89].

- A Tamper Resistance-based module deters unauthorized changing of a system by preventing alterations, or preserving evidence of alteration[90].

- The Third Party Validation pattern is intended to provide validation of messages broadcasted in a given network. If a spoofed message is sent to a node and the receiving node cannot validate the message with a trusted node somewhere else in the network, then the receiving node can disregard the message [83].

4.4 Sample Automotive Design Patterns

The following two sections describe two sample patterns, one based on inward-facing communication and another for outward-facing communication. The pattern *Authorization* is the example for inward-facing, while the pattern *Blacklist* is for the outward-facing example. Underlined text refers to pattern names, in-line italics refers to UML diagram elements (participants), and courier font refers to security principles/guidelines. The complete set of automotive cybersecurity patterns are included in Appendix A

4.4.1 Authorization

4.4.1.1 Pattern Name and Classification

The Authorization pattern is a Structural Security Pattern.

4.4.1.2 Intent

Authorization provides a structure that facilitates access control to resources.

4.4.1.3 Motivation

Many systems need to restrict access to their resources according to certain criteria (e.g. a security policy) [64]. In automotive systems, inadequate authentication and authorization protocols have exposed the system to a variety of security exploits such as attacks on inter-vehicle networks [36], spoofing ECUs [39], and various other exploits [38, 60]. In the case of Cranchelli et al. [1], the motivation for developing an authentication protocol for a communications bus, similar to a CAN bus, comes from the potential for ECUs and ECU messages to be spoofed. According to SAE standard J3061, **preventing unauthorized access to data** as well as controlling component privileges are key principles in developing automotive cybersecurity for a system [2]. According to Viega McGraw [73] the principle of being **reluctant to trust** improves the systems abilities to mitigate any security exploits a sub-system may have. This principle is particularly applicable to an automotive system, as many components come from different parties, and increasingly, vehicles have more communication with external entities.

4.4.1.4 Properties

The *Authorization* can be used to satisfy the Authentication property and the Authorization property.

4.4.1.5 Applicability

Authorization is applicable to attack Prevention.

4.4.1.6 Structure

The *Authorization* structure of a system can be captured in terms of their relationships (Figure 4.1) by a domain model using UML class diagram notation, the boxes represent domain elements connected by relationships, where a dashed line relates properties of the relationship (e.g., rights of an authorization). Active entities are represented by instances of a *Subject* class and passive resources by instances of a *Protection Object* class. Between those main participants exists a relation that portrays which *Subject* is entitled to access certain objects. The properties of this relationship are organized in the association class *Rights*. The objects of this class define the type of access, transfer conditions, and constraints that restrict the use of *Rights*.

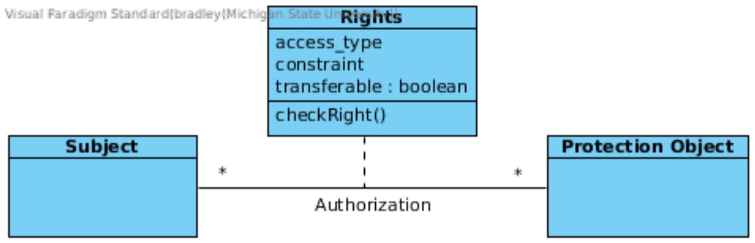


Figure 4.1 Class Diagram for Authentication Pattern

4.4.1.7 Participants

The following section describes the participating classes in the pattern structure.

- *Protection Object*
  - Represents passive resources of the system that are accessed by *Subjects*.
- *Rights*
  - Defines the properties of the authorization rule between *Subject* and *Protection Object*.
  - The access type property describes which kind of access of the current *Rights* object grants. Commonly, this property holds values in the style of read, write, execute.
  - The constraint property is a predicate that describes under which circumstances the current *Rights* object is valid and may grant a certain privilege.
  - The transferable property determines whether a right is transitive and its connected *Subject* may grant the right to other active entities.
- *Subject*

– Represents active entities that need to access *Protected Objects* in accordance to their *Rights*.

4.4.1.8 Collaborations

Different *Subjects* in the system want to access *Protection Objects*. In order to make use of a certain resource they need to request access to it from the responsible controlling instance. This instance will check if an association class between the *Subject* and the *Protected Object* exists that justifies the required access request. Depending on this examination, access is granted or not.

4.4.1.9 Behavior

A UML sequence diagram is used to capture behavior in systems, where object entities have interactions for messages indicating changes of behavior. The solid arrows indicate system messages and dashed arrows indicate acknowledged messages. A *Subject* wishing to access a *Protected Object* will request access at a *Checkpoint Object* (Figure 4.2). The *Checkpoint* will request the rights pertaining to the *Subject*, and if approved, will forward the request to the *Protected Object.*

4.4.1.10 Constraints

In an automotive system, example constraints on an Authorization pattern include the performance of the authorization protocol and the amount of resources it may require. Because a vehicle is limited in the resources it has available (e.g.. CPU and memory), there is a need to perform

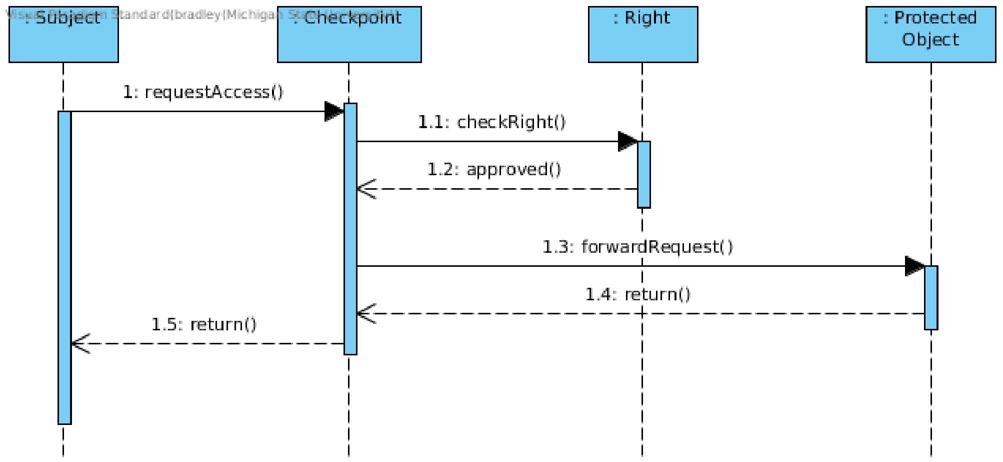


Figure 4.2 Sequence Diagram for Authentication Pattern

authorization efficiently. In addition, the need for real time execution; where multiple *Subjects* may be sharing access to a *Protected Object*, delay is often unacceptable.

4.4.1.11 Consequences

Table 4.3 describes consequences in implementing the pattern described in terms of performance perspectives, such as confidentiality, availability, and manageability.

4.4.1.12 Known Uses

As mentioned in the motivation section, Cranchelli [1] has developed an authorization module for a shared communication bus to authenticate and manage authorized communication between ECUs on the bus [1]. It is able to authenticate ECUs on the bus at system start-up, and by using digital signatures, authorize ECUs' interactions with one another.

4.4.1.13 Supported Principles

The Authorization pattern should be used in combination with Principle 4 (**principle of least privilege**) to assign the user's rights. Furthermore, Authorization can be used to **compartmentalize** (Principle 5) the system and reduce the impact of a security breach. For example, an attacker that illegally manages to authenticate as user A, has only the rights user A would have. Given a restrictive security policy, the enforcement of access rights **promotes privacy** (Principle 7).

Table 4.3 Consequences for Authorization Pattern

Accountability:	Not affected.
Confidentiality:	Can be improved by specification and rigorous enforcement of rights.
Integrity:	Can be improved by specification and rigorous enforcement of rights.
Availability:	Can be improved by specification and rigorous enforcement of rights.
Performance:	The system performance might be derogated by extensive right checks and the evaluation of the rights constraint predicates. This issue is of particular concern in a real-time environment.
Cost:	Cost is not significantly affected by the application of this pattern. May require additional hardware to perform authentication.
Manageability:	Modify to account for automotive systems e.g., ECUs, communication privileges, etc. instead of traditional computing terminology.
Usability:	If the access rights are checked extensively the user might recognize a loss of performance. Authorization may limit utilization of shared resources, impede safety-critical sub-systems.

4.4.1.14 Related Security Patterns

The *Check Point* pattern can be used to examine requests by using the structure of the Authorization pattern. The *RBAC* or Roles pattern is an extended version of this pattern.

4.4.2 Blacklist

4.4.2.1 Pattern Name and Classification

Blacklist is a structural based security pattern.

4.4.2.2 Intent

A Blacklist intends to monitor the traffic of potentially malicious addresses in a network. Nodes in the network use the Blacklist to block traffic originating from the malicious nodes.

4.4.2.3 Motivation

Communication with external entities is becoming increasingly integrated into modern automotive systems. Today's cars communicate with smart infrastructure, receive over the air updates from manufacturers, and exchange information with other vehicles on the road. While these technologies allow for improved performance and user experience, the increased reliance on networks leaves a vehicle susceptible to attack from a malicious user on the network. The inability to identify compromised nodes poses a security risk to an automotive system. VANETs allow vehicles and smart infrastructure to share information about current driving conditions within a specific broadcast range. Due to the routing protocols upon which VANETs rely, a malicious node can join a VANET and proceed to attack the network by intercepting the communication between vehicles, dropping packets, and changing or fabricating packets[91]. Applications such as the Post Crash Notification [92] can be deployed on VANETs to detect an attack such as injections of false messages that can lead to potentially dangerous responses by the vehicles on the network.

4.4.2.4 Properties

The Blacklist can be used to satisfy the Authentication property, the Authorization property, and the Non-Repudiation property.

4.4.2.5 Applicability

A Blacklist is applicable to the Prevention and Mitigation of an attack.

4.4.2.6 Structure

The Blacklist structure of a system can be captured in terms of their relationships (Figure 4.3). An entity sending a message is captured by a *Client* object. The entity secured with a Blacklist is represented as a *Service* object. The interface between the two objects is a *Checkpoint* object. And finally, the Blacklist captured with the *Blacklist* object.

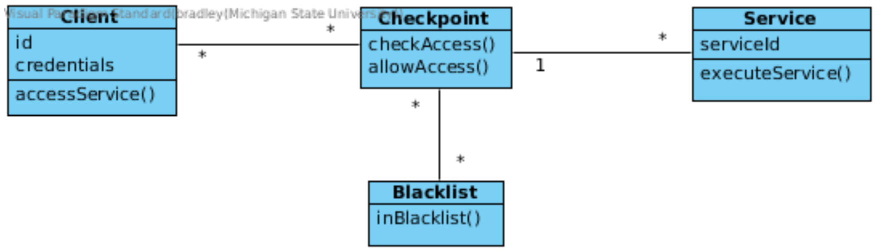


Figure 4.3 Class Diagram for Blacklist Pattern

4.4.2.7 Participants

The following section describes the participating classes in the pattern structure.

- *Client:* An actor requesting access to a node, or *Service* object.
- *Service:* The intended protected object. A *Service* has access to a system's message processing.
- *Blacklist:* The system's list object for tracking bad addresses.
- *Checkpoint:* The interface through which all communication with a *Service* is achieved, and where the *Blacklist* is checked.

4.4.2.8 Collaborations

A given *Client* will attempt to communicate with a *Service* through a *Checkpoint*. Consequently, there is an association between both the *Service* and the *Checkpoint* and the *Client* and the

Checkpoint that acts as the interface. There is also an association between a *Checkpoint* and the associated *Blacklist*.

4.4.2.9 Behavior

A Blacklist (Figure 4.4), is a solution that can partially fulfill the relevant security requirements that are unmet in the problem statement. A Blacklist maintains a list of addresses within a network that have exhibited inappropriate behavior. When a packet from a blacklisted address arrives at a node, the node simply drops the packet. In the context of a VANET, nodes will not process or forward any messages that originate from a blacklisted address, and routing algorithms will not include the blacklisted nodes in calculating routes of the packets.

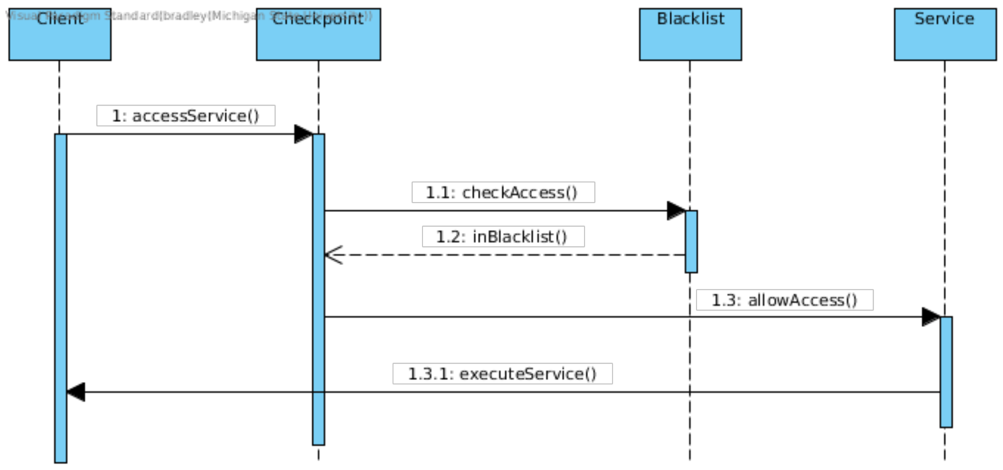


Figure 4.4 Sequence Diagram for Blacklist Pattern

4.4.2.10 Constraints

Real-time constraints are important to consider in the context of the Blacklist pattern as message processing incurs overhead as well as consumes additional resources per message.

4.4.2.11 Consequences

See Table 4.4.

4.4.2.12 Known Uses

An example of a Blacklist being deployed in an automotive setting is given by Daeinabi and Rahbar [81]. Here the authors propose a system in which a select number of nodes in a VANET

Table 4.4 Consequences of Blacklist Pattern

Accountability:	Preventing misbehaving nodes from participating in a network improves the ability of the network to hold malicious actors accountable.
Confidentiality:	Preventing nodes that are known to misbehave from accessing data sent between nodes in the network provides a higher degree of data confidentiality.
Integrity:	Preventing nodes that are known to misbehave from receiving and possibly modifying data sent between nodes in the network provides a higher degree of data integrity.
Availability:	Depending on the blacklisting protocol the Blacklist may prevent harmless nodes from participating in the network, thereby reducing availability.
Performance:	Performance may be affected by the overhead incurred by using the network Blacklist pattern. Performance improvements may occur however as the resources consumed by misuse are reduced .
Cost:	Not applicable.
Manageability:	By setting blacklist rules, manageability of a network can be improved.
Usability:	Depending on the blacklisting protocol, legitimate users may be prevented from accessing the services.

are tasked with monitoring behavior of the other nodes, where the overall network can dynamically change over time. If one of the monitoring nodes detects abnormal traffic from a given node, then it increases a distrust value for that node. Meanwhile, all nodes in the network maintain a Blacklist. If a node is given a distrust value beyond a certain threshold by the monitoring nodes, then that node is reported and blacklisted for the other nodes in the network.

4.4.2.13 Relevant Security Principles

The Blacklist leverages Practice Defense in Depth, Reluctance to Trust, and Compartmentalize.

4.4.2.14 Related Patterns

The Blacklist is related to traffic filtering patterns such as the Firewall pattern and with another pattern the Signature-Based IDS.

CHAPTER 5

SCP-AC: SITUATIONAL CRIME PREVENTION FOR AUTOMOTIVE CYBERSECURITY

The connectivity capabilities of modern vehicles, such as the telematics system, GPS, and vehicle-to-vehicle communication have developers and researchers working with these systems to create technical solutions for securing autonomous vehicle's constraints. This chapter presents a model-based SCP framework that combines crime theory with technical solutions to provide a socio-technical, stakeholder-centered approach to address automotive cybersecurity.

Research focused on securing self-driving vehicles has largely targeted specialized technical solutions to accommodate the constraints specific to the autonomous vehicle domain [10, 11]. However, limited research has explicitly accounted for the range of stakeholders, each of which expose different vulnerabilities based on their role with the inward and outward communication. The stakeholders include technicians working with the vehicles, the software developers, and the consumer themselves. The SCP framework from criminology theory is used to categorize different prevention strategies for a given crime in a respective environment; it was originally developed for traditional crime (e.g., theft or physical attack) [7]. While the SCP has been customized for general and specialized crime (e.g., physical crimes such as burglary and vandalism, as well as cybercrime on an enterprise system [3]), it has not been applied to the automotive cybersecurity domain and its unique constraints.

This chapter introduces SCP-AC, a model-based situational crime prevention framework developed to address automotive cybersecurity. The SCP-AC framework is a domain-focused method to enable a human-centered technology-enabled prevention strategy for addressing automotive cybercrime by extending the environment of a crime and accounting for the entire lifecycle of the vehicle. The SCP-AC framework also explicitly categorizes a number of strategies for preventing a crime based on the roles and activities applicable to the range of stakeholders. Our contributions expand the strategies of crime prevention from a specific crime in an isolated environment to all stakeholders involved across the duration of an ongoing attack. We expand cybersecurity by not only looking at how and why a crime occurred, to focusing on who is involved and what they could do differently to prevent an attack. We also combine different prevention strategies based on their usage, either in research (i.e., state-of-the-art) or in industry practices (i.e., state-of-the-practice) to help inform researchers and industry collaborators.

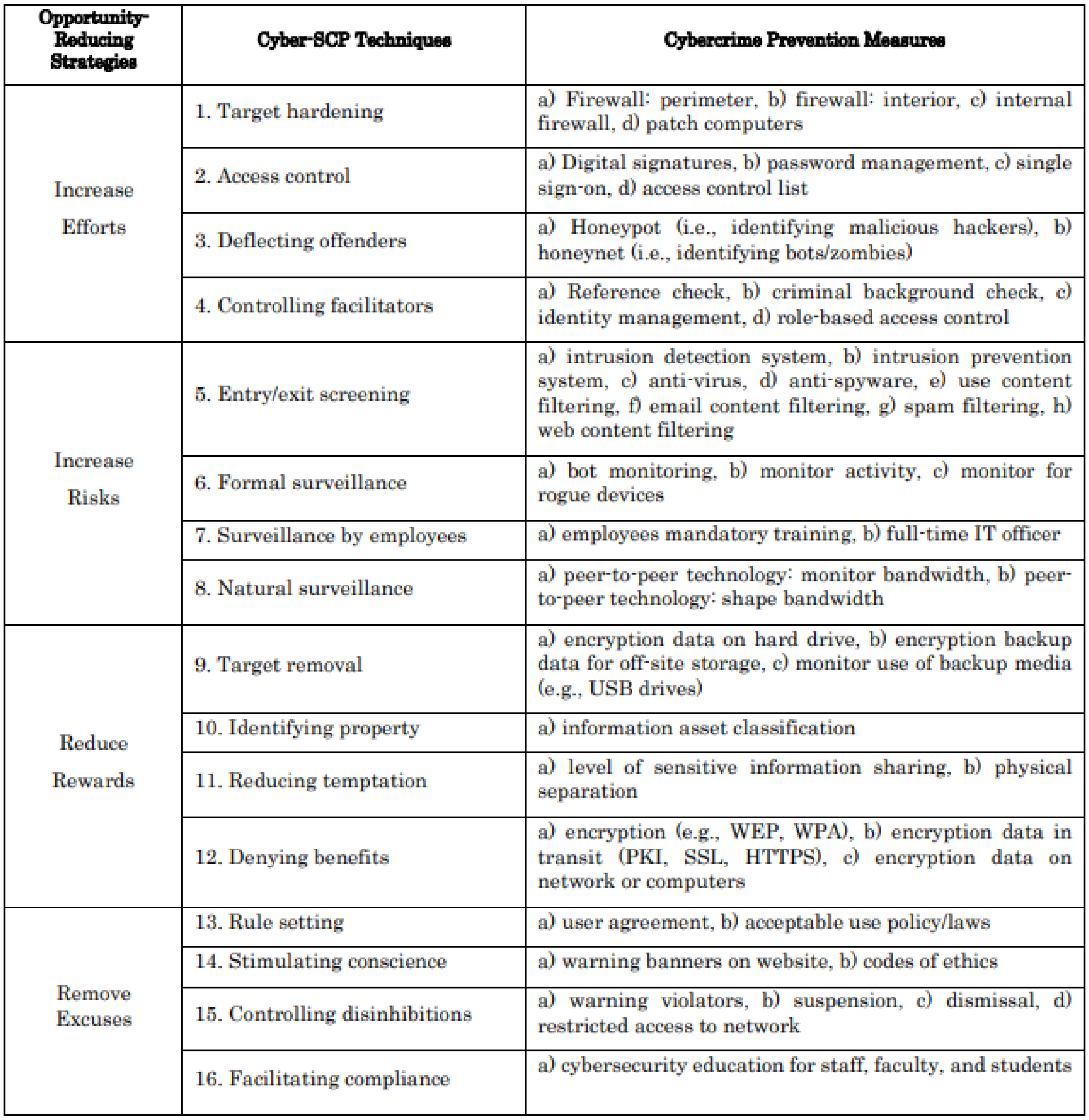
The SCP-AC framework covers a range of prevention strategies for varying stakeholders interacting within the automotive domain. The SCP-AC framework can be adapted for relevant techniques specific to autonomous vehicles, while also supporting a range of stakeholders. The model is also accessible when needing to find more information on a given solution. These strategies and potential threats are analyzed through a newly created socio-technical threat assessment approach, focusing on human factors, such as degree of access (e.g., physical vs. remote) needed to perform the attack and the impact on safety. The strategies leverage domain expertise and feedback from industrial collaborators (e.g., OEMs and suppliers) on where techniques are being used, either in a research setting or in current industry practices [28]. The SCP-AC relational model integrates crime prevention, threat assessment, and prevention strategies to enable a diverse range of solutions inside automotive vehicles. By synergistically combining the structure of the SCP-AC, industry input, and a criteria for categorizing problems and solutions, we have a systematic method for analyzing threats and prevention strategies for autonomous vehicles.

In order to create an accessible interface for all users, a web-based interface was created to support a wide range of queries for different types of users. The SCP-AC framework and relevant repository of information (i.e., threat surfaces, threat to stakeholder) can be queried by keywords that reference threat vectors, types of attacks, stakeholder involved, onboard subsystems, prevention strategies, etc. The SCP-AC framework can also be used to identify hot spots, ones that require more urgent attention from the research community and/or educating the different stakeholders.

5.1 The SCP-AC Matrix

The cybercrime SCP matrix [3] attempts to modify different prevention categories from the original SCP matrix in order to better accommodate the cyber-domain. Through this modification, the matrix is better suited for showing domain-specific solution strategies inside the cyber-space

Table 5.1 The Cyber-Situational Crime Prevention Techniques [3]



Opportunity-Reducing Strategies	Cyber-SCP Techniques	Cybercrime Prevention Measures
Increase Efforts	1. Target hardening	a) Firewall: perimeter, b) firewall: interior, c) internal firewall, d) patch computers
	2. Access control	a) Digital signatures, b) password management, c) single sign-on, d) access control list
	3. Deflecting offenders	a) Honeypot (i.e., identifying malicious hackers), b) honeynet (i.e., identifying bots/zombies)
	4. Controlling facilitators	a) Reference check, b) criminal background check, c) identity management, d) role-based access control
Increase Risks	5. Entry/exit screening	a) intrusion detection system, b) intrusion prevention system, c) anti-virus, d) anti-spyware, e) use content filtering, f) email content filtering, g) spam filtering, h) web content filtering
	6. Formal surveillance	a) bot monitoring, b) monitor activity, c) monitor for rogue devices
	7. Surveillance by employees	a) employees mandatory training, b) full-time IT officer
	8. Natural surveillance	a) peer-to-peer technology: monitor bandwidth, b) peer-to-peer technology: shape bandwidth
Reduce Rewards	9. Target removal	a) encryption data on hard drive, b) encryption backup data for off-site storage, c) monitor use of backup media (e.g., USB drives)
	10. Identifying property	a) information asset classification
	11. Reducing temptation	a) level of sensitive information sharing, b) physical separation
	12. Denying benefits	a) encryption (e.g., WEP, WPA), b) encryption data in transit (PKI, SSL, HTTPS), c) encryption data on network or computers
Remove Excuses	13. Rule setting	a) user agreement, b) acceptable use policy/laws
	14. Stimulating conscience	a) warning banners on website, b) codes of ethics
	15. Controlling disinhibitions	a) warning violators, b) suspension, c) dismissal, d) restricted access to network
	16. Facilitating compliance	a) cybersecurity education for staff, faculty, and students

and identifying techniques to better secure cybercrime systems. This matrix focuses on the different solutions for preventing cybercrime from occurring for a number of different attacks. See Table 5.1.

However, by leveraging the cybercrime SCP matrix in Table 5.1 and revising it to focus on the automotive cybersecurity domain and its relevant stakeholders, we created the SCP-AC shown in Table 5.2. The SCP-AC matrix focuses on the intersection between the stakeholders and technical aspects of the vehicle. The solutions, and the relevant applicability will vary depending on which stakeholder is interacting with given aspects of the vehicle. Therefore, this SCP-AC needs to holds an array of information for each entry, specifying the solution strategy, the domain where this strategy takes place, the relevant stakeholder, and the attack being prevented. This can be seen in the example showing a single category for the SCP-AC, that is increasing the effort in Table 5.3. This template will be used for all entries inside our SCP-AC, shown in this format, [Strategy, Domain, Stakeholder, Attack, Reference]. An example of an entry inside our matrix is as follows: [Firewall, On-Board, Customer, DDoS, [[93]]], indicating this in-depth information is needed as the stakeholder for a given solution can change, and certain solutions may not be applicable across every aspect of the autonomous vehicle. For example, keeping in-depth records about who is accessing specific areas of the autonomous vehicle may be necessary for developers, but may not help security if applied to a customer. These changes also apply to the domain that varies across autonomous vehicles, and can range from the physical car to an over-the-air update system. These distinctions need to be specified for autonomous vehicles to help identify solutions, and specify which aspect of this socio-technical system needs to be addressed. The sample of the given matrix can be seen below in Table 5.2.

5.2 Categorizing Templates

The different strategy types, for example increasing the effort, require a more in-depth approach that should be reusable and based on different stakeholders working across the domain of self-driving vehicles. The goal is to either: increase the effort, increase the risk, reduce the rewards of the attacker, or remove excuses from varying stakeholders, where the scope of the vehicle extends

Table 5.2 SCP-AC: Overview

Increase the Effort	**Increase Risks**	**Reduce Rewards**	**Remove Excuses**
Target Hardening	Entry/Exit Screenings	Target Removal	Rule Setting
Access Control	Formal Surveillance	Identifying Property	Stimulating Conscience
Deflecting Offenders	Surveillance by Employees	Reducing Temptations	Controlling Disinhibition
Controlling Facilitators	Natural Surveillance	Denying Benefits	Facilitating Compliance

Table 5.3 SCP-AC: Increase the Effort

Target Hardening
[Firewall, On-Board, Customer, DDoS, [93]]
Access Control
[Entry/Exit Screening, Development, Developers, Privilege Escalation, [94]]
Deflecting Offenders
[Honeypot, On-Board, Developers, Resource Access, [95]]
Controlling Facilitators & Natural Surveillance
[Market Restrictions, Everywhere, Customers / Employees, System Compromises, []]

far past the vehicle itself, and interacts with different types of stakeholders and domains. As such, the utilization of our threat assessment, different fields, and current on-going research in the field of autonomous vehicles enable us to create a foundation and a system for identifying, categorizing, and creating prevention strategies.

5.3 Using the SCP-AC Matrix

The SCP-AC matrix can be used as a categorization for solution strategies on how to secure these autonomous vehicles, it can also be used as a framework for analyzing risks and determining if a prevention strategy is feasible for a given part of the vehicle. Before identifying a specific threat or risk, a given aspect of this autonomous vehicle system can be examined through the scope of the SCP-AC to determine if any of these strategies can be used to further bolster defense against a threat. This SCP-AC was then used to create a database for organizing and searching for relevant solutions based on a number of criteria. Additionally, data mined information from relevant forums was inserted into the database to allow that information to be queried for more in-depth analysis. Through this approach, aspects of the vehicle can be made more secure before exploits, threats, or attacks are executed.

5.3.1 SCP-AC Search System

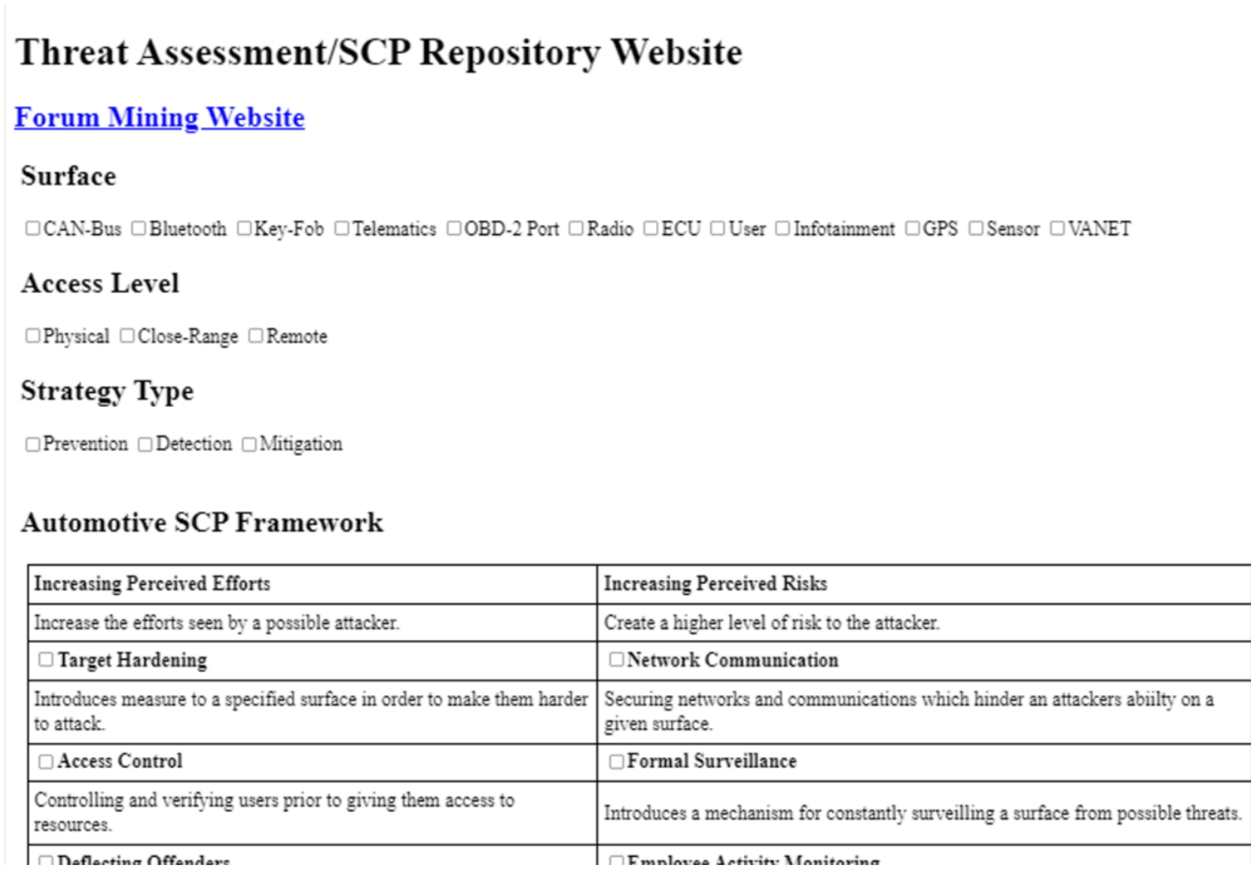
The SCP-AC we created is used in tandem with the vector-specific threat assessment to support a database and query system for finding targeted solutions to documented cybersecurity threats. This database serves as a repository of the documented solutions found in research literature. The system can be searched using a number of criteria. The first criteria to filter results was based on the relevant threat vector, for example a CAN-Bus, Bluetooth, or ECU. Next, the user can search based on the level of access required for the attack, either physical, close-range, or remote. The solution type is also searchable, using prevention, detection, or mitigation. We then incorporated our SCP-AC with a brief description to enable the user to filter strategies by either category inside the SCP-AC or by a specific type such as increasing efforts or risk. Lastly, the user can choose what information is displayed based on all the above criteria or simply entering a keyword. Figure 5.1 displays the web-based interface for searching the SCP-AC repository, which enables a user to select the threat surface(s), distance to target, and type of solution (i.e., prevention, detection, and mitigation). Figure 5.2 displays sample search results, indicating hyperlinks to relevant solutions. Through this system, relevant threats and solutions are made more accessible to developers working on the implementation or creation of cybersecurity techniques.

5.3.2 Forum Mining

The SCP-AC search system was also extended to support querying results mined from a number of online forums related to automotive cybersecurity. The results were gathered and processed by Holt et al. [96] to uncover potential automotive cybersecurity vulnerability information from popular and prevalent enthusiasts websites. This information can be leveraged to identify new and emerging threats or hot spots to be investigated. Search can be filtered based on the specific forum website, whether it was a hardware or software related issue, and which threat vector. Furthermore, manufacturer-specific information also enabled the user to search based on type of vehicle. Lastly, the user has the ability to query based on keywords (e.g TPMS, telematics). Figure 5.3 displays the forum mining interface.

5.4 State of the Practice vs. State of the Art

Through our collaboration with Ford Motor Company [28], we gained input and feedback from industry collaborators and were able to distinguish between prevention techniques used in practice (i.e., currently in industrial use) or theoretical (i.e., in research). We defined these differences in techniques as state-of-the-practice, the techniques currently in use in the automotive industry. Also state-of-the-art techniques, these that were researched and implemented in a research setting but may not be implemented across the automotive industry. These differences helped us to categorize approaches and organize solutions for future work based on type of usage.


Threat Assessment/SCP Repository Website

Forum Mining Website

Surface

☐CAN-Bus ☐Bluetooth ☐Key-Fob ☐Telematics ☐OBD-2 Port ☐Radio ☐ECU ☐User ☐Infotainment ☐GPS ☐Sensor ☐VANET

Access Level

☐Physical ☐Close-Range ☐Remote

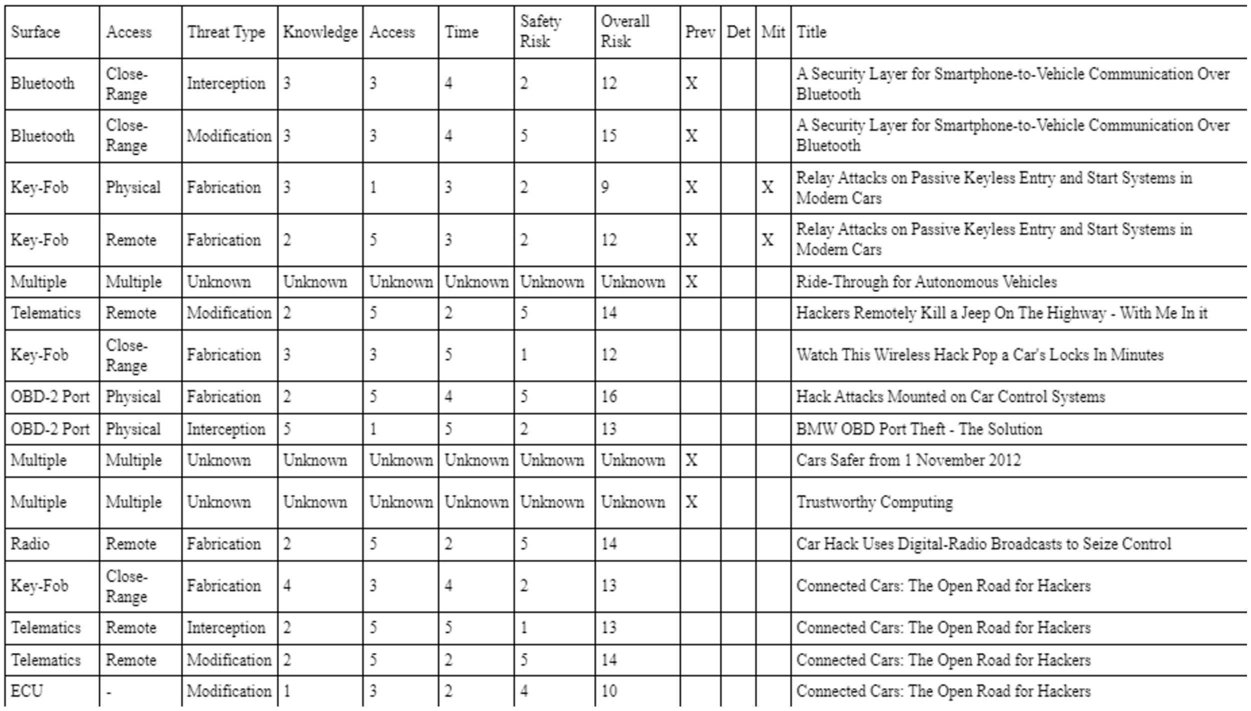
Strategy Type

☐Prevention ☐Detection ☐Mitigation

Automotive SCP Framework

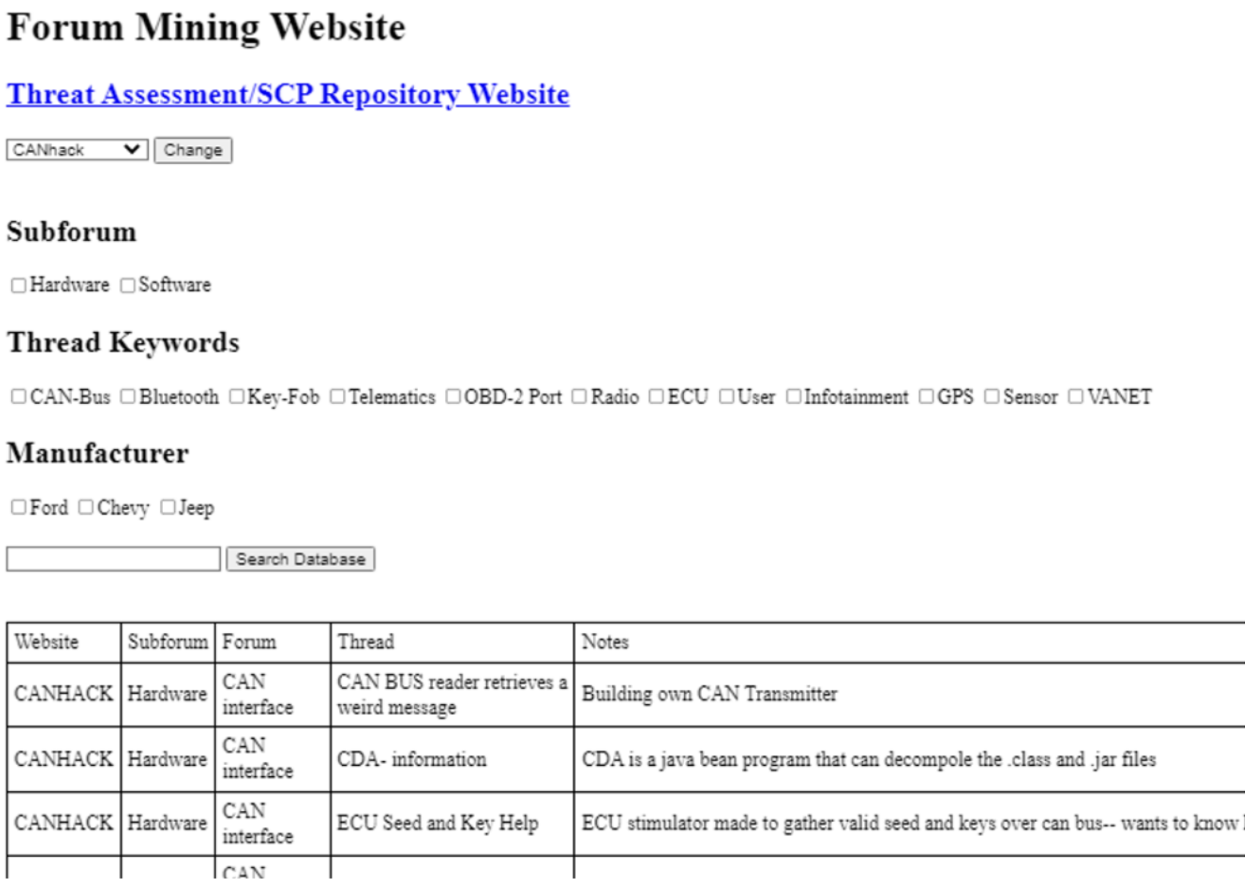
Increasing Perceived Efforts	Increasing Perceived Risks
Increase the efforts seen by a possible attacker.	Create a higher level of risk to the attacker.
☐ **Target Hardening**	☐**Network Communication**
Introduces measure to a specified surface in order to make them harder to attack.	Securing networks and communications which hinder an attackers abiilty on a given surface.
☐ **Access Control**	☐**Formal Surveillance**
Controlling and verifying users prior to giving them access to resources.	Introduces a mechanism for constantly surveilling a surface from possible threats.
☐ **Deflecting Offenders**	☐ **Employee Activity Monitoring**

Figure 5.1 SCP-AC: Assessment Search



Surface	Access	Threat Type	Knowledge	Access	Time	Safety Risk	Overall Risk	Prev	Det	Mit	Title
Bluetooth	Close-Range	Interception	3	3	4	2	12	X			A Security Layer for Smartphone-to-Vehicle Communication Over Bluetooth
Bluetooth	Close-Range	Modification	3	3	4	5	15	X			A Security Layer for Smartphone-to-Vehicle Communication Over Bluetooth
Key-Fob	Physical	Fabrication	3	1	3	2	9	X		X	Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars
Key-Fob	Remote	Fabrication	2	5	3	2	12	X		X	Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars
Multiple	Multiple	Unknown	Unknown	Unknown	Unknown	Unknown	Unknown	X			Ride-Through for Autonomous Vehicles
Telematics	Remote	Modification	2	5	2	5	14				Hackers Remotely Kill a Jeep On The Highway - With Me In it
Key-Fob	Close-Range	Fabrication	3	3	5	1	12				Watch This Wireless Hack Pop a Car's Locks In Minutes
OBD-2 Port	Physical	Fabrication	2	5	4	5	16				Hack Attacks Mounted on Car Control Systems
OBD-2 Port	Physical	Interception	5	1	5	2	13				BMW OBD Port Theft - The Solution
Multiple	Multiple	Unknown	Unknown	Unknown	Unknown	Unknown	Unknown	X			Cars Safer from 1 November 2012
Multiple	Multiple	Unknown	Unknown	Unknown	Unknown	Unknown	Unknown	X			Trustworthy Computing
Radio	Remote	Fabrication	2	5	2	5	14				Car Hack Uses Digital-Radio Broadcasts to Seize Control
Key-Fob	Close-Range	Fabrication	4	3	4	2	13				Connected Cars: The Open Road for Hackers
Telematics	Remote	Interception	2	5	5	1	13				Connected Cars: The Open Road for Hackers
Telematics	Remote	Modification	2	5	2	5	14				Connected Cars: The Open Road for Hackers
ECU	-	Modification	1	3	2	4	10				Connected Cars: The Open Road for Hackers

Figure 5.2 SCP-AC: Data Sample


Forum Mining Website

Threat Assessment/SCP Repository Website

CANhack | Change

Subforum

☐ Hardware ☐ Software

Thread Keywords

☐ CAN-Bus ☐ Bluetooth ☐ Key-Fob ☐ Telematics ☐ OBD-2 Port ☐ Radio ☐ ECU ☐ User ☐ Infotainment ☐ GPS ☐ Sensor ☐ VANET

Manufacturer

☐ Ford ☐ Chevy ☐ Jeep

Search Database

Website	Subforum	Forum	Thread	Notes
CANHACK	Hardware	CAN interface	CAN BUS reader retrieves a weird message	Building own CAN Transmitter
CANHACK	Hardware	CAN interface	CDA- information	CDA is a java bean program that can decompole the .class and .jar files
CANHACK	Hardware	CAN interface	ECU Seed and Key Help	ECU stimulator made to gather valid seed and keys over can bus-- wants to know
		CAN		

Figure 5.3 SCP-AC: Forum Mining

CHAPTER 6

SOCIO-TECHNICAL AUTOMOTIVE SECURITY DESIGN PATTERNS

The wide range of human-based interactions from consumer to employees provide not only attackers with additional stakeholders to target as a source of vulnerabilities, but also provide new opportunities for stakeholders (including the broader community) to play an altruistic role in securing the system. This chapter presents work that integrates criminology theory with model-based development to create reusable socio-technical security design patterns, explicitly focusing on the role of human behavior in automotive cybercrime prevention.

Research focusing on securing self-driving cars has targeted technical solutions with specific software constraints [97, 98, 99], but given the increasing vulnerabilities accompanying the fast-paced advancements with autonomous vehicles it is strategic to broaden the community of contributors and the scope of automotive cybersecurity solutions to leverage strategies from other disciplines and related application areas [100]. Criminology concepts within the field of cybersecurity have also been applied to cyber-physical system to create more informative attack profiles [101]. Security improvements have also been made to systems by educating the community and leveraging informed user behavior, such as detecting and responding to phishing within phone services [25]. The use of design patterns, or reusable problem-solution pairs, have helped address cybersecurity vulnerabilities relevant to autonomous vehicles [15]. Previous work on automotive cybersecurity patterns focused on one specific stakeholder, namely, technical solutions that the developer can use to address automotive cybersecurity vulnerability [15].

This chapter introduces socio-technical design patterns to leverage stakeholder behavior and interactions to address automotive cybersecurity. This work contributes two key insights. The first insight is that by taking a stakeholder-focused view on automotive cybersecurity, we can develop complementary solution strategies to technical-based solutions, where stakeholders can play different roles with actions that directly or indirectly address a given cybersecurity vulnerability. While humans may be identified as an attacker, we also consider stakeholders where they serve a complementary role in preventing, discovering, and/or reporting automotive cybersecurity vulnerabilities. The stakeholders can include technicians at dealerships, software engineers working with various subsystems, and passengers. Our second key insight is that by taking a stakeholder-focused view of solutions, we are able to generalize the solution strategies to apply at a meta-level, thereby making them applicable to multiple automotive cybersecurity vulnerabilities. For example, different ethical hacker [102] events can be tasked to discover different (types of) vulnerabilities. The crime theory terminology and conceptual framework provide a means for us to identify and model the stakeholders relevant to the cybersecurity vulnerabilities.

The purpose of the socio-technical design patterns is to provide reusable problem-driven cybersecurity solution strategies for automotive systems that use direct or indirect actions taken by stakeholders complementary to the traditional techniques used by developers. The problem (i.e., vulnerability) is described in terms of criminology concepts to capture the routine of stakeholders with various roles related to an attack. The solution strategy is described in terms of socio-technical prevention tactics. The stakeholders included in the attacks can have direct or indirect impacts on cybersecurity for a system, either with the attack itself or in preventing the attack from occurring. Each of the socio-technical design patterns capture stakeholder behavior that can be instantiated and applied to handle multiple cybersecurity vulnerabilities at different stages of the autonomous vehicle lifecycle. This approach is intended to engage a broader community of stakeholders who can help address automotive cybersecurity vulnerabilities.

The set of socio-technical design patterns have been informed and developed from a variety of sources such as state of the art literature [99, 103], industry best practices, and collaborations with criminology experts [100]. We gratefully acknowledge contributions from Thomas Holt [96] and Jay Kennedy [104]. Additionally, our collaborators from the Michigan State University School of Criminal Justice and collaborations with Ford Motor Company [28].

6.1 Design Structure

We use the Unified Modeling Language (UML) class diagrams [17] to visually depict stakeholders and their routines that contribute to crime. Specifically, this includes the opportunity and occurrence of a crime surrounding a previously identified cybersecurity vulnerability. We use aspects of the crime to define the crime triangle stakeholder, mainly to focus on their specific routines. The key stakeholder (*target* and *offender*) have a given routine with their set of actions that are taken regularly in accordance with a *environment* relevant crime. However, due to an autonomous vehicle's expansive set of different *environments* (e.g., subsystems) that the attacks can take place, it is necessary to draw explicit attention to identifying the routine and actions taken in that respective *environment*.

Next, we are able to use the relationships visualized via the class diagrams to identify the stakeholders that can change behavior to improve security measures. By using information from the crime triangle, we explicitly model entities that may exist and have a level of control over various stakeholders (i.e., managers), specifically those outlined in the crime triangle (*handler*, *guardian*, *caretaker*). Those entities are able to use prevention strategies from the SCP framework to make changes to the various routines of our stakeholders.

6.2 Attack Definition

The problem description is outlined using the previously discussed set of attack characteristics that can be used to assess the security and impact of an attack. The use of other threat identification models (e.g., MITRE [105]) could be incorporated after vehicle deployment with a repository of threats/attacks. *The type of cybersecurity attack* is identified as either an interception (e.g., eavesdropping), interruption (e.g., ransomware attack), fabrication (e.g., generating false data), or modification attack (e.g., removing or changing sensor data) [106]. The type of attack and how it is perpetrated can help in identifying the key stakeholders and the assets that may be targeted. The next characteristic is the *degree of safety compromised*, ranging from low, medium, or high. This characteristic can help stakeholders determine the urgency/priority for addressing the attack. Securing as many aspects as possible is the goal, however time and resources must be carefully allocated in order to secure autonomous vehicles by the most dangerous threats first. Next, *the distance between offender and target* helps stakeholders determine what kind of access is needed to commit an attack. The range can be anywhere from physical, close-range, or remote. This helps in identifying who is capable of making changes to a given system, and what strategies may be more effective than others. Lastly, *the time required to complete an attack* (low, medium, high) is the last detail that can also assist in identifying the severity of an attack. The asset and target may be in various levels of danger depending on how quickly the attack for the autonomous system can be executed.

6.3 Template

This section presents the template that we developed to describe the socio-technical design patterns.

6.3.1 Pattern Name and Classification

The pattern name provides a description of the solution strategy with a classification as either structural or behavioral. A structural pattern is classified by changes to the organization or policies for stakeholders. A behavioral pattern focuses on changes in behavior of a stakeholder.

6.3.2 Intent

This section describes the functional intent of the pattern, what purpose it will serve for addressing a problem, and what solution strategy will be used to prevent attacks (including applicable SCP categories).

6.3.3 Also Known As

This describes other known keywords or aliases for a specified pattern.

6.3.4 Motivation

This field describes the vulnerability under consideration and its problem context, including the roles played by the relevant stakeholders.

6.3.5 Applicability

This section describes what aspects of security are being addressed, specifically prevention, detection, and/or mitigation. The goal should be to address prevention, but some can be extended to perform other aspects of security for autonomous vehicles (e.g., mitigating an attack by training victims to respond to attacks to minimize impact).

6.3.6 Participants

This field describes entities participating in the design pattern (i.e, data dictionary).

Crime: Details of the Crime that is being committed within an Environment, and must include an Offender, a Target, and an Attack.

Target: Target of a Crime, including a description.

Routine: This represents actions a given entity is taking on a regular basis. The Routine can be either passive or active The Routine can be defined for a Target, Offender, Environment, Caretaker, Handler, or Guardian. The methods can include a security method or other specific actions taken by a given stakeholder.

Guardian: This class represents an entity with a level of control over the Target. They can have fields and methods specifying their impact over the Target, altering the Routine, monitoring the Target, or make changes to the Target.

Offender: This represents the Offender with a description of the types of offender who may attack the system.

Handler: This field describes an entity who has a level of control over the Offender. The Handler can include methods to change the Offender or alter the Routine.

Attack: This represents details of an Attack used by the Offender. The type (interruption, interception, fabrication, or modification), the degree of knowledge (low, medium, or high), the distance required (physical, close-range, or remote), and an amount of time for execution (low, medium, high). The methods can include how they access the asset or attack specific operations (e.g., monitor).

Vulnerability: Describes the exploit that is used to successfully execute an Attack to gain access to the Target. This may also include details about the Vulnerability.

Environment: This represents the Environment where the crime takes place. The environment can include fields such as name, relevant methods such as those that monitor or manage.

Caretaker: This describes an entity who can help protect the Environment. The Caretaker can take actions that include altering the Environment, monitoring, and changing the routine.

SituationalCrimePrevention: This represents the Situational Crime Prevention strategy being applied, with a description of each category type and the strategy being used. Situational-Crime Prevention can propose changes to a managing entity (Caretaker, Guardian, Handler). The type of SituationalCrimePrevention can be either Increase Effort, Increase Risk, Reduce Rewards, or Remove Excuses.

6.3.7 Collaborations

Interactions between the participant and the role each participant plays in the design pattern.

Crime: The crime is committed using an Attack by an Offender, within an Environment, and happens to a Target.

Target: This is Target of the Crime which can be protected by a Guardian. The Target may use a specific Routine.

Routine: Set of actions used by a given stakeholder for a pattern. Furthermore, the Routine applies information from the
SituationalCrimePrevention field to change stakeholder Routine.

Guardian: The Guardian protects the Target and may use a Routine.

Offender: Performs a Crime by using an Attack. They can be controlled by a Handler if present, and may use a Routine

Handler: Controls the Offender and may utilize a Routine.

Attack: The Attack exploits a Vulnerability on the Target to commit a Crime.

Vulnerability: The Vulnerability is exploited by an Attack to gain access to the Target

Environment: The Environment is where the Crime occurs, can be affected by a Caretaker, and may have a Routine.

Caretaker: The Caretaker is responsible for the Environment, and may have a Routine.

SituationalCrimePrevention: The SituationalCrimePrevention applies to a Routine to make changes to a given stakeholder.

6.3.8 Structure

This section uses the UML class diagram to capture the structure of the pattern (see Figure 6.1). The elements are described by the entities from the participants field (3.3.6) and their interactions described in the collaborations field (3.3.7).

6.3.9 Consequences

This field describes socio-technical consequences with a focus on stakeholder actions when the pattern is implemented (see Table 6.1). The constraints are also included, which cover either legal, hardware related, or physical constraints that may be imposed on the solution strategy.

6.3.10 Implementation

This section outlines relevant techniques, types, and variations of the pattern that can be implemented.

6.3.11 Known Uses

This field discusses examples of this pattern's use in an automotive setting, directly or indirectly.

6.3.12 Related Patterns

This section describes any relevant security patterns that may be related (e.g., security [15, 64, 78]).

6.4 Overview of Socio-technical Patterns Repository

The following gives a high-level description of each of our socio-technical design patterns to date.

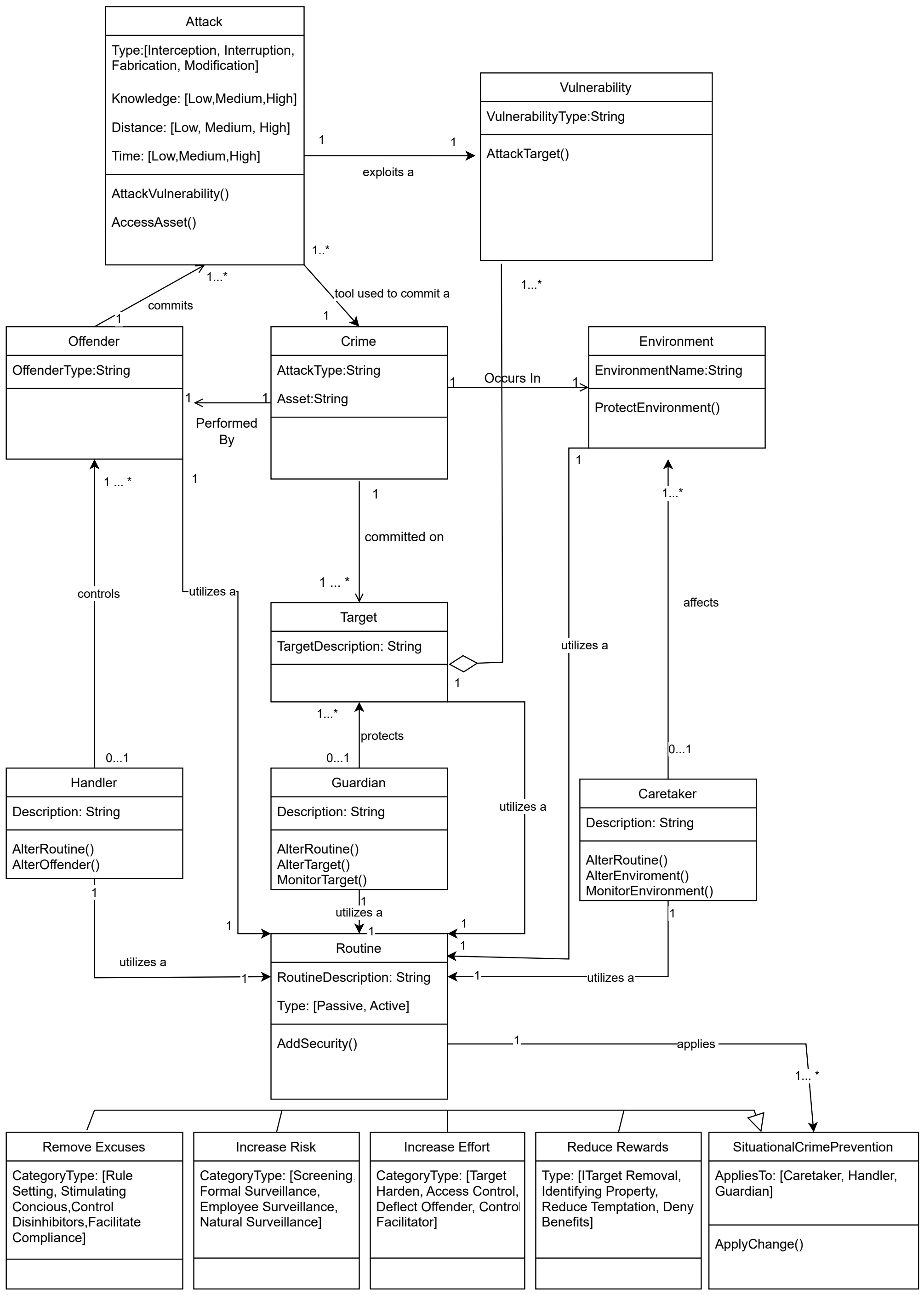


Figure 6.1 Socio-technical Security Design Pattern Template Structure

Table 6.1 Consequences for a Pattern.

Consequence	Description
Accountability:	Participating stakeholders need to be accountable for their behavior with respect to the design pattern objectives (including any resulting policies).
Confidentiality:	Impact on confidentiality of artifacts and/or stakeholder information when pattern is implemented (e.g., proprietary data for a subsystem)
Integrity:	The stakeholder information and relevant facts should not be compromised after pattern implementation (e.g., vehicle credentials maintained).
Availability:	The ability of a stakeholder to access a resource needed to realize a pattern (e.g., logging in with credentials, time availability).
Performance:	The ability to execute key functions by a stakeholder after pattern changes are made (e.g., employee completing task).
Cost:	The increase in maintenance or implementation associated with implementation of a pattern (e.g., monetary cost, time for policy changes).
Manageability:	Impact on supervising a system or users by a stakeholder (i.e., enforcing new policies).
Usability:	The ability to correctly operate a function by a stakeholder after design pattern produces deliberate changes (e.g., policy recommendation).

Bug Exploit Discovery: This pattern uses incentives (e.g., rewards) to encourage online users to report any discovered cyber (security) flaws in a system. The purpose is to assist organizations responsible for the development and assurance of automotive systems in finding exploitable bugs while encouraging the broader community to take more altruistic approaches to address/prevent cybersecurity vulnerabilities.

Tool Restriction: This seeks to restrict high-risk tools used in automotive vehicle attacks by eliminating them from circulation or by monitoring who is purchasing them (e.g., key for signal listeners). The strategy is to reduce accessibility of the tool, as well as make vendors aware of the illegal use of that given tool.

Gameout: This helps vehicle owners by educating them in how to respond to high-risk situations where attacks can be perpetrated throughout an autonomous vehicle (e.g., manipulating infotainment system [107]). The purpose of this pattern is to educate users about attack information, and encourage a safe set of actions to be taken to stop attacks from occurring successfully.

Crowdsourcing with Experts: This creates a method for successfully organizing and executing crowdsourcing events (such as hack-a-thons) to enlist experts in the community to help in identifying vulnerabilities for an autonomous vehicle. The purpose of this pattern is to leverage technical skills from experts in the field to play the role of an “attacker" (i.e., ethical hackers [102]). This can help identify different vulnerabilities in a system and help identify areas that can then be fixed and strengthened.

Public Resource Security: This focuses on securing publicly accessible resources that interact/interface with the autonomous vehicle, such as wireless connectivity at a dealership, over-the-air updates, in-vehicle WiFi, and Bluetooth connectivity. The purpose of this pattern is to create social behavior mechanisms (e.g., guidelines, policies, standards) for protecting publicly-accessible resources that can be used for direct or indirect attacks.

Social Engineering Protection: This involves training autonomous vehicle drivers and other stakeholders such as employees against social engineering attacks (phishing) through education of cybersecurity information and sample attacks. The purpose is to prevent various types of phishing or social engineering attacks by educating drivers, the employees at dealerships, or with developers working on autonomous vehicle-related software.

Forum Monitoring: This pattern enlists the use of moderators or managers of forum websites to discourage discussion of illegal or illicit activities related to autonomous vehicles. The purpose is to encourage these moderators to remove information that is intentionally purposed to exploit automotive cybersecurity vulnerabilities to do harm. In addition, the moderators should discourage discussions and/or share information relating to automotive cybersecurity attacks.

Table 6.2 gives an overview of each of the proposed patterns. It describes applicability, whether the strategy seeks to focus on detection, prevention, and mitigation. Table 6.2 also shows the relevant SCP strategy for the pattern, such as increasing effort, increasing risk, reducing rewards, or

Table 6.2 Applicability and SCP Strategy of Select Patterns

Pattern Name	Applicability	+ Effort	+ Risk	- Rewards	- Excuses	Prevention Goal	Stakeholder
Bug Exploit Discovery	Prev/Det				X	Encourage Bug Reporting	Community Users
Tool Restriction	Prev/Det	X			X	Restrict Malicious Tool Sales	Third Party Vendors
Gameout	Prev/Det/Mit				X	Train Vehicle Owners	Vehicle Owners
Crowdsourcing with Experts	Prev/Det			X	X	Discover Vulnerability	External Experts
Public Resource Security	Prev				X	Secure Public Points (Wifi, Bluetooth)	Resource Managers
Social Engineering Protection	Prev/Det		X		X	Phishing Protection for Developer	Developers
Forum Monitoring	Prev/Det/Mit		X			Find Inquiries and Discussions for Potential Threats	Public Forum Moderators

removing excuses. Lastly, the main prevention goal is outlined to summarize the key socio-technical solution being proposed and the key relevant stakeholders for applying the patterns.

For these patterns the goal is to balance enough information about respective stakeholders and their responsibilities, but remain abstract enough to provide users with reuse. The patterns include a "Known Uses" field to provide information about application.

6.5 Sample Patterns

The following patterns have been selected to illustrate our socio-technical framework. The collection of the socio-technical patterns can be found in Appendix B.

6.5.1 Crowdsourcing with Experts

The Crowdsourcing with Experts pattern leverages expert's (i.e., stakeholders in the community who have specialized expertise) technical skills in the field of autonomous vehicles to expose vulnerabilities in a controlled setting. This pattern enables an organization to complement their in-house cybersecurity capabilities with a broader community of expertise to identify vulnerabilities and inform their follow-on security protection mechanisms, as well as applicable security policies.

6.5.1.1 Pattern Name and Classification

The Crowdsourcing with Experts pattern is a structural security pattern due to the policy changes needed for organizing this type of event. The policies must also relate to the level of access or internal (non-public) information to event participants.

6.5.1.2 Intent

Crowdsourcing with Experts provides a pattern for enabling users to search for vulnerabilities during events such as a hack-a-thon. The goal is to have highly-skilled experts objectively analyze threat vectors to identify potential vulnerabilities. It is important for event organizers to find a balance between information access and exposing vulnerabilities. This pattern can use *Removing Excuses* and *Reduce Rewards* to create policies and clear strategies to conduct a successful Crowdsourcing with Experts event. The pattern can *reduce temptations* by incentivizing (via monetary or altruistic reasons) users to take safer actions and *rule setting* clear policies surrounding positive behaviors. This can be seen in Table 6.3, indicated by the blue entries.

Table 6.3 SCP Strategies Applied for Crowdsourcing with Experts

Increase the Effort	**Increase Risks**	**Reduce Rewards**	**Remove Excuses**
Target Hardening	*Entry/Exit Screenings*	*Target Removal*	Rule Setting
Access Control	*Formal Surveillance*	*Identifying Property*	*Stimulating Conscience*
Deflecting Offenders	*Surveillance by Employees*	Reducing Temptations	*Controlling Disinhibition*
Controlling Facilitators	*Natural Surveillance*	*Denying Benefits*	*Facilitating Compliance*

6.5.1.3 Also Known As

This is similar to using third-party groups to improve the security of the system, including crowdsourced testing [108], crowdsourced funding [109], crowdsourced labor [110], or idea crowdsourcing [111].

6.5.1.4 Motivation

This pattern uses experts in the field to help identify and exploit vulnerabilities through an organized event [112]. The solution to this problem entails providing system-specific artifacts (e.g., proprietary or only internally available) to a (diverse) group of external experts to harness their individual and/or collective expertise and domain knowledge to identify potential vulnerabilities within a given system.

6.5.1.5 Applicability

Addresses attack Prevention and Detection.

6.5.1.6 Participants

The following describes the participants in the pattern structure.

CompromiseSystem: The CompromiseSystem is the crime being perpetrated by EthicalHacker in the Vehicle. The AttackType is the exposure and successful attack on a vulnerability. The Asset will be defined in the parameters of the event by the EventHost.

Subsystem: The Subsystem is the target for the event. The Subsystem has a function, and also has a number of connected devices.

SubsystemRoutine: The SubsystemRoutine represents either a passive or active set of actions taken by the Subsystem. The Subsystem can execute the core functionality, and if the system are in place prevent and detect cybersecurity threats.

Vehicle: This represents the Vehicle or environment with information about the make, model, and year.

VehicleRoutine: The set of passive or active actions taken inside the Vehicle. This can include securing the vehicle itself, preventing attacks, detecting attacks, and mitigating attacks.

EventHost: This represents the EventHost or manager who is hosting the crowdsourcing event. The EventHost will have a budget of money allocated as well as a time limit for the event itself. The EventHost will also decide on an invite policy for participants (e.g., open or selective), what type of event (e.g., issue-oriented or tech oriented), and where the event is hosted (e.g., virtual, in-person, hybrid) [112]. The event can also be competitive or collaborative.

EventHostRoutine: Represents the passive or active actions taken by the EventHost. This can include giving information to the EthicalHacker, monitoring the Subsystem, or monitoring the Vehicle.

EthicalHacker: This hacker uses a level of knowledge to expose and break a vulnerability in order to help prevent future attacks.

EthicalHackerRoutine: Set of active actions taken by hacker when searching for vulnerabilities that compromise the system and expose information.

HackerAttack: This represents the details of a HackerAttack that is used. The HackerAttack can be any type of attack (interception, interruption, fabrication, modification), the attack may requires low, medium, or high level of knowledge from the EthicalHacker. It can be executed from a distance (low, medium, and high), and will require low to medium time to execute due to the time constraints of the event. The goal of the HackerAttack is to exploit the system and focus on the specified asset.

PreviouslyUnknownVulnerability: This represents the type of PreviouslyUnknownVulnerability that will be used in the crime of CompromiseSystem. The location will be discovered during the crowdsourcing event, and can only expose the Target.

RemoveExcuses: The RemoveExcuses is an SCP strategy that sets rules. The rules are set by adding a policy, system focus, and rule for how the crowdsourcing events are conducted.

ReduceRewards: The ReduceRewards is an SCP strategy that lowers the desire to attack a system by reducing the projected reward. This is done by hosting crowdsourcing events and making a systems defenses more public to reduce the desire to attack.

6.5.1.7 Collaborations

This section describes the different collaborations occurring within the system for different elements.

CompromiseSystem: The CompromiseSystem is performed by the EthicalHacker in the Vehicle. The CompromiseSystem occurs using the tool of a HackerAttack to attack the Subsystem.

Subsystem: The Subsystem is the target of the CompromiseSystem. The Subsystem is protected by the EventHost.

SubsystemRoutine: Actions taken by the Subsystem.

Vehicle: The context where the CompromiseSystem occurs.

VehicleRoutine: Set of actions taken inside the Vehicle.

EventHost: This represents the EventHost that affects the Vehicle, protects the Subsystem, and manages the EthicalHacker.

EventHostRoutine: Actions taken by the EventHost.

EthicalHacker: The EthicalHacker is CompromiseSystem on a Subsystem within a Vehicle. The EthicalHacker uses a type of HackerAttack to CompromiseSystem.

EthicalHackerRoutine: Actions taken by the EthicalHacker.

HackerAttack: This represents the HackerAttack that is used to CompromiseSystem. It exploits a PreviouslyUnknownVulnerability.

PreviouslyUnknownVulnerability: This represents the type of PreviouslyUnknownVulnerability that will be exploited by the type of HackerAttack. This is part of the Subsystem.

RemoveExcuses: The RemoveExcuses applies to EventHostRoutine.

ReduceRewards: The ReduceRewards applies to EventHostRoutine.

6.5.1.8 Structure

The relevant UML class diagram can be found in Figure 6.2 for the Crowdsourcing with Experts pattern. The data dictionary for the diagram has been described by Participants and Collaborations field to describe the elements and their routines, respectively.

6.5.1.9 Consequences

The constraints of the pattern are having numerous experts discovering hidden vulnerabilities inside the vehicle. The issues found are dependent on the group's success in finding threat vectors, the time constraints allocated for the event, and access to information. Table 6.4 describes consequences in applying the pattern.

Table 6.4 Consequences for Crowdsourcing with Experts

Consequence	Description
Accountability:	The EthicalHacker is responsible for finding vulnerabilities within the Subsystem.
Confidentiality:	The number of EthicalHackers who can access proprietary parts of Vehicle.
Integrity:	The EthicalHacker may be required to sign a non-disclosure agreement.
Availability:	EthicalHacker access to the Vehicle.
Performance:	The EventManager needs to provide adequate resources for EthicalHackers to complete their task.
Cost:	The costs to host crowdsourcing event by the EventManager.
Manageability:	EthicalHacker need to be managed and observed (e.g., information given and obtained.)
Usability:	N/A

6.5.1.10 Implementation

This event can be done through a physical event, a virtual event, or a hybrid event. The hosting of a hack-a-thon can be selective or open when recruiting, and be competitive or collaborative in nature. The level of information divulged to the group of ethical hackers can vary. This can be issue- or technology-oriented [112].

6.5.1.11 Known Uses

The design pattern of Crowdsourcing with Experts can be used to gain valuable insight into a systems operation. For example, within information systems, 242 programs and 6095 white hat

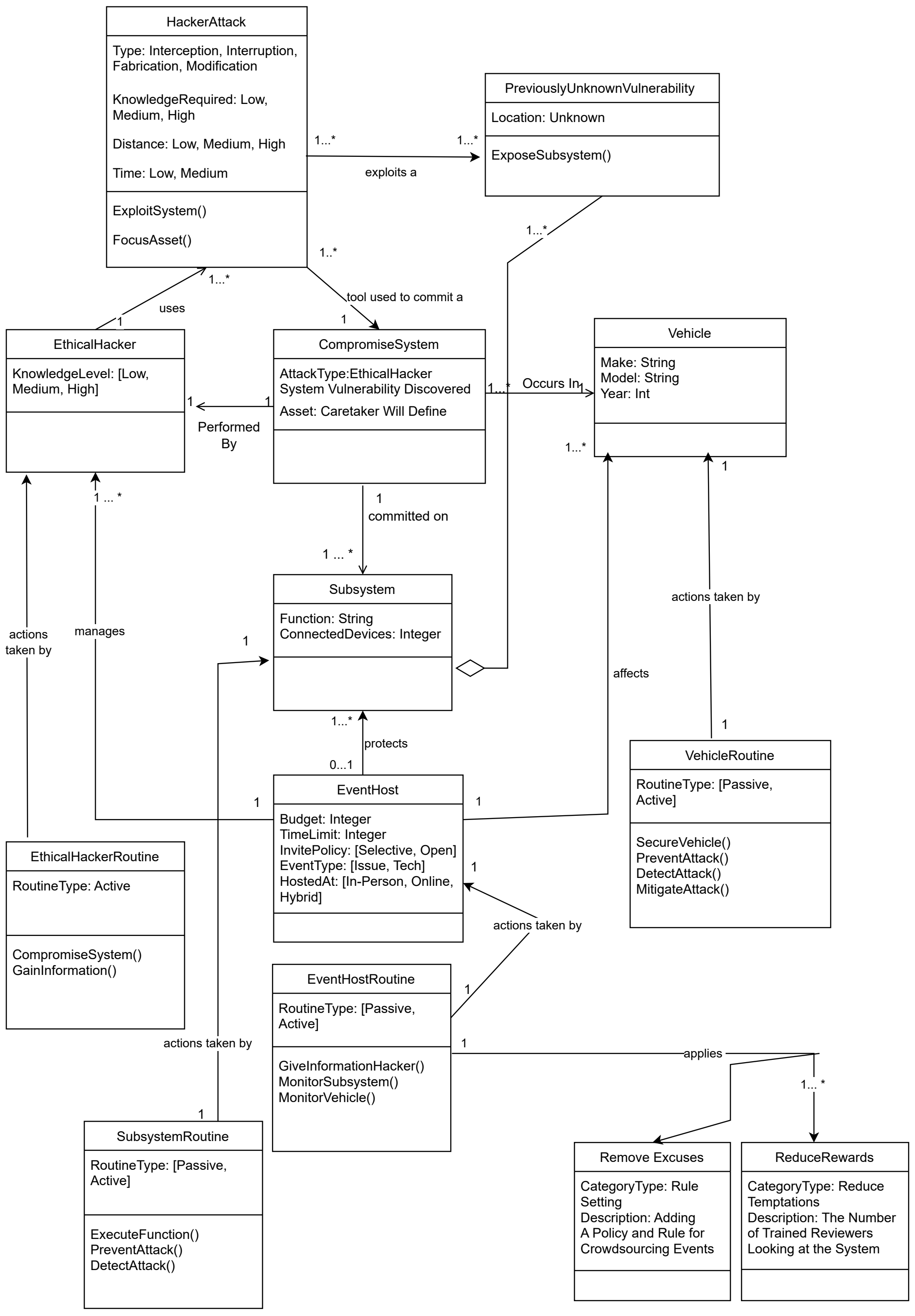


Figure 6.2 Crowdsourcing with Experts

hackers data was collected and analyzed to find information, such as outcome control, expressing commitment, and communication [113]. This was done to analyze feasibility of white-hat hackers. It was found that crowdsourcing can increase the submission quantity of vulnerabilities [113]. The work done for Crowdsourcing with Experts are all performed by experts using legal actions that are monitored by the host of the crowdsourcing event.

6.5.1.12 Related Patterns

The Crowdsourcing with Experts pattern could be used in conjunction with the Public Resource Security pattern to find vulnerabilities and require changes to secure a system.

6.5.2 Social Engineering Protection

The Social Engineering Protection pattern intends to prevent social engineering or phishing attacks on employees by exposing and educating users on the methods of those types of attacks. Through user education, attacks will be more difficult to penetrate systems via the employees and provide stronger cybersecurity to the system.

6.5.2.1 Pattern Name and Classification

The Social Engineering Protection pattern is a behavioral security pattern due to the behavioral aspect of monitoring user responses to events. Furthermore, through education the behavior of users is trying to be modified.

6.5.2.2 Intent

Social Engineering Protection provides a pattern for testing and educating users within a system on avoiding social engineering or phishing attacks. The goal is to educate users and identify weak points in stakeholders operating inside a company. Once the information of susceptible users is available, specialized training can be added and required to help create a more robust working environment for stakeholders. The specific techniques would vary based on susceptible users, type of testing employed, and education levels. The SCP strategies implemented can be seen in Table 6.5.

6.5.2.3 Also Known As

This is known as phish testing, phishing sampling, or phishing education of users [114, 115, 116].

Table 6.5 SCP Strategies for Social Engineering Protection

Increase the Effort	**Increase Risks**	**Reduce Rewards**	**Remove Excuses**
Target Hardening	*Entry/Exit Screenings*	*Target Removal*	*Rule Setting*
Access Control	*Formal Surveillance*	*Identifying Property*	*Stimulating Conscience*
Deflecting Offenders	*Surveillance by Employees*	*Reducing Temptations*	*Controlling Disinhibition*
Controlling Facilitators	*Natural Surveillance*	*Denying Benefits*	*Facilitating Compliance*

6.5.2.4 Motivation

The problem being addressed by the Social Engineering Protection pattern is social engineering, phishing, and user manipulation of stakeholders on self-driving vehicles. The solution strategy addresses prevention and detection methods for identifying vulnerable stakeholders inside a system, then educating necessary individuals to strengthen defense of the system itself.

6.5.2.5 Applicability

Social Engineering Protection is applicable to attack Prevention and Detection.

6.5.2.6 Participants

The following field describes the participating classes in the pattern structure.

StealingCredentials: This represents the crime of StealingCredentials that can be used to gain access to information or other credentials. This can be leveraged for further, more damaging attacks.

User: This represents the target of a StealingCredentials attack that focuses on User to complete the attack. The User can have a level of social engineering knowledge.

UserRoutine: This represents the routine of a User that focuses on the actions taken. This will have either an active or a passive routine type, and can do things such such as connect device, login, manage accounts, and browse the internet.

Workplace: This represents the environment where the crime takes place, in this case the Workplace. The Workplace may have attributes such as a location, number of employees, and number of connected devices.

WorkplaceRoutine: This represents the routine of the Workplace. The WorkplaceRoutine can take actions, including providing emails and providing internet.

Manger: This entity is the caretaker and guardian for both the Workplace and the User. They will have a description, and can take actions to secure the entities it is responsible for.

ManagerRoutine: The set of actions taken by the Manger. This can be either a passive or active routine, and will try to enhance the security by training users. identifying high-risk users, or monitoring emails.

Phisher: This represents the offender of a crime as a Phisher with a description of their knowledge level and the ability to bulk email Users.

Phishing: This represents the details of an attack that is a Phishing used by the Phisher. The type of attack can be fabrication, interception, or modification. It will require a medium to low to medium knowledge, and distance along with time will be determined based on the attack. The goal with the attack is to focus on a vulnerability and access the described asset.

UntrainedUser: This is the UntrainedUser that requires a User and perform a given Phishing attack on.

IncreaseEffort: This is the SCP strategy of IncreaseEffort that will make the target more difficult to attack. This can be done by educating users to spot the warning signs of a phishing attack.

6.5.2.7 Collaborations

This describes the relationships between entities.

StealingCredentials: The crime of StealingCredentials requires a User to attack a Phisher inside the Workplace. The Phishing attack is the manner in which the StealingCredentials is performed.

User: This represents the target of an StealingCredentials on a User. The User has a set of actions and is managed by a Manger

UserRoutine: This represents the set of actions of a User.

Workplace: The environment where the StealingCredentials takes place, in this case the Workplace. The Workplace has a set of actions and is managed by the Manger.

WorkplaceRoutine: This represents the set of actions of a Workplace.

Manager: This is the caretaker and guardian for both the Workplace and the User. The Manager has a set of actions.

ManagerRoutine: This represents the set of actions of a Manager.

Phisher: The entity committing the StealingCredentials using a Phishing.

Phishing: The Phishing is used by the Phisher to perform the StealingCredentials crime. This also exploits a UntrainedUser.

UntrainedUser: This is the UntrainedUser vulnerability on a User and perform a given Phishing attack.

IncreaseEffort: This SCP strategy should be applied to the Manager.

6.5.2.8 Structure

The relevant class and sequence UML diagrams can be found in Figure 6.3 for the Social Engineering Protection pattern.

6.5.2.9 Consequences

The constraints of the Social Engineering Protection pattern are allocating resources for phishing and hosting education seminars. The full information can be seen in Table 6.6.

6.5.2.10 Implementation

This can vary depending on preference, but both a detection mechanism for testing and an educational component are both necessary.

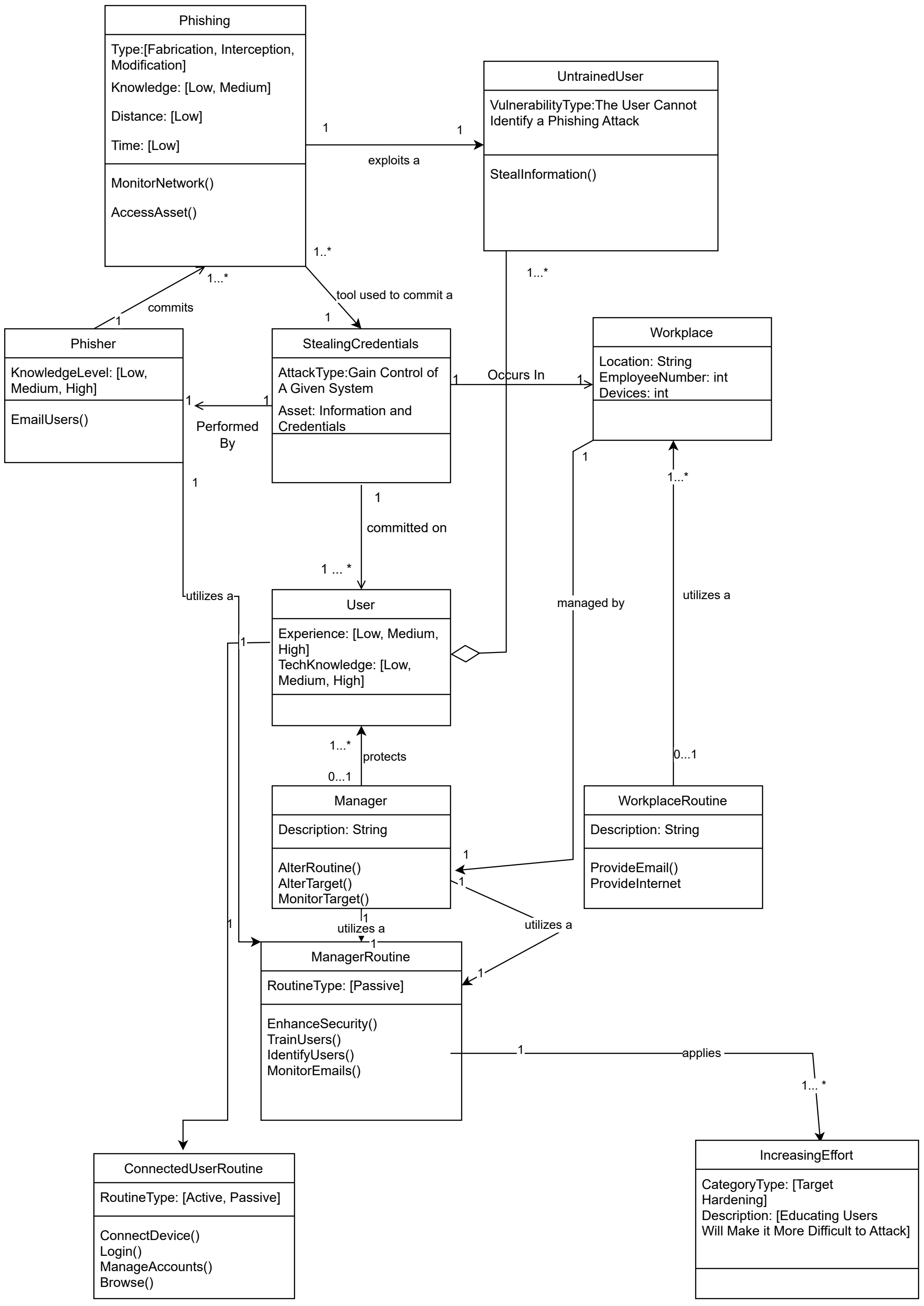


Figure 6.3 Social Engineering Protection

Table 6.6 Consequences for Social Engineering Protection

Accountability:	The Manager needs to implementing and observe required testing and education.
Confidentiality:	N/A
Integrity:	N/A
Availability:	The time required to train Users would be altered.
Performance:	N/A
Cost:	This may increase cost to have educational seminars taking away from workflow.
Manageability:	The number of people who need to be managed and monitoring would increase.
Usability:	N/A

6.5.2.11 Known Uses

There are a variety of different testing mechanisms for identifying vulnerable stakeholders, but issues with users become bias towards testing cases can cause issues. CyberPhishing [117] is a technique that attempt to merge "in the wild" attacks with testing attacks to provide better testing results for stakeholders. Outside of seminar, presentations, or traditional learning techniques for avoiding phishing attacks some phone applications have been developed [118].

6.5.2.12 Related Patterns

Gameout

CHAPTER 7

CONCLUSION AND FUTURE WORK

The research addressing cybersecurity for modern automotive vehicles has largely focused on software-based and technical solutions [11, 12, 13]. The technical solutions are intended to secure the automotive system prior to the vehicles being deployed to the public. However, additional measures can be incorporated to address cybersecurity vulnerabilities by focusing solutions around the different stakeholders who directly or indirectly contribute to the vehicles operation (e.g., dealership, users, maintenance). A socio-technical solution can be explored in order to add additional measures to the cybersecurity issue. The use of a stakeholder-based preventative strategies, with the goal of addressing the human factors and behavior, provide another dimension of cybersecurity prevention for automotive systems.

Our work has proposed solution strategies for securing self-driving vehicles with different levels of human involvement. We began with a detailed assessment of cybersecurity threats around self-driving vehicles. We created a custom assessment tool to help identify cybersecurity vulnerabilities of the vehicle using metrics such as time required to perform an attack, skills required to execute, and type of access to the given component. This assessment was used to inform our subsequent research and solution strategies to address automotive cybersecurity.

Next, we developed automotive security design patterns that focus on the onboard technical prevention strategies for autonomous vehicles, with the developer as the main target stakeholder. These strategies were created by leveraging our previous threat assessment tool to help identify problem areas in cybersecurity for autonomous vehicles. Through reusable design patterns, developers have strategies such as, authorization, firewall, and blacklisting to help secure self-driving vehicles.

Then we developed the SCP-AC framework to address automotive cybercrime prevention analysis from the perspective of multiple stakeholders, considering both human-based and technical-based prevention strategies. By organizing prevention strategies into categories such as increasing effort, increasing risk, and removing excuses for multiple stakeholders, both new and existing strategies can be created and/or modified to provide complementary approaches that require different types of effort and potentially yield different results. Using the SCP-AC, we are now able to categorize and develop solutions from both a technical and a human-based perspective to automotive cybercrime.

Finally, we created our socio-technical automotive security design patterns with an emphasis on roles played by different stakeholders. By using UML, we were able to create a stakeholder-based domain model that integrates criminology concepts and the different actors within a domain. The domain model was included as part of the template that was developed to describe socio-technical design solutions focused on prevention from a stakeholder-based perspective. The template then described a set of patterns that can be instantiated to leverage a broader community of stakeholders to address cybersecurity risks in automotive cybersecurity.

The overarching objective of this work has been to present three complementary socio-technical solutions that differ in the stakeholder(s) involved and the balance between technical and social-based solutions. The design pattern work created reusable problem/solution pairs to technical solutions to address cybersecurity for autonomous vehicles intended to be used by the different stakeholders. Next, we sought to integrate more social and stakeholder-based solutions by creating an organizational matrix called the SCP-AC. This matrix was leveraged to categorize and identify solutions strategies that already exist or can be proposed in the future need to expand to represent stakeholders involved. Lastly, we created socio-technical design patterns to focus on solution strategies for stakeholders from the broader community (beyond developers) to help secure self-driving cars. By covering a range of technical and social solutions, we are able to create a cybersecurity prevention framework that addresses two major factors at work, the technology and the stakeholders interacting with them.

7.1 Summary of Contributions

There were three major objectives to this research:

1. To analyze and rank threats to automotive security through a human-focused perspective.

2. To utilize crime theory concepts to offer additional and alternative perspectives to securing vehicles.

3. To create reusable, stakeholder-focused solutions for securing automotive systems.

With these research objectives in mind, this dissertation makes the following research contributions:

- Identify and categorize automotive vulnerabilities, cybersecurity attacks, and solutions across a modern vehicle includes a threat assessment to categorize circumstances around an attack [14].

- Create a repository of reusable automotive cybersecurity design patterns to address cybersecurity vulnerabilities in the automotive system, these patterns are technical in nature and intended for developers of automotive systems. This included both inward-facing and outward-facing communication [14, 15].

- Create and organize different solutions using situation crime prevention strategies to focus on stakeholder involvement with varying degrees of human interaction to prevent cybersecurity attacks [16].

- Develop a template that was used to create a repository of stakeholder-based socio-technical design solutions to address cybersecurity for automotive systems [19]. These patterns are socio-technical and intended for a broader community of stakeholders, including dealerships, third-party vendors, hobbyists, etc.

7.2 Future Work

The work described in this dissertation can be leveraged and extended along multiple dimensions for further investigations. The following topics are possible directions to pursue.

Socio-technical Design Patterns for Supply Chain: The socio-technical pattern template can be extended to develop additional sample patterns for stakeholders within the supply chain domain. The number of stakeholders operating inside the environment (e.g., consumer, retailer, supplier, manufacturer) working with technology leaves the system at risk if one aspect were to fail. The flow of goods, materials, and operations inside the supply chain require all aspects to work at a specific, uninterrupted rate. This creates a high-risk situation that leaves the supply chain vulnerable to shutdown if an aspect is negatively affected or targeted by an attack. The problem being addressed would be identify, create, and secure proper prevention strategies to ensure stakeholders are able to work without negatively impacting other elements of the supply chain system.

Medical Field Socio-technical Design Pattern Application: The socio-technical design pattern approach can be applied to other cyber-physical systems and disciplines. Specifically, many advanced medical technologies interact directly with patients or physicians (e.g., Da Vinci robotic surgery arm [119], Bluetooth-enabled pacemakers). The stakeholders working with medical technology on developing, managing, and operating leave attackers with different ways to exploit a system. The nature of medical, cyber-physical systems lends itself well to stakeholder-focused solutions as humans are interfacing with these technologies. The problem being addressed is the prevention and security of different resources, tools, or equipment to ensure proper precautions are handled in securing these systems. Furthermore, ensuring the safety of users with no interruption of service is crucial to developing a safe ecosystem of medicine for cyber-physical devices.

Interdisciplinary Criminology Work Extension: The application of traditional crime techniques to cybersecurity can be leveraged for testing, in addition to added prevention techniques. The goal of analyzing motivations and tendencies around traditional crime systems can have applications to help secure systems involved in a number of fields. The use of rational-choice theory [120] discusses a persons response to commit a crime based on estimated risk/reward. The idea that attackers may act on opportunity of a reward while weighing risks [120] can be applied to various systems to help make them more robust. Furthermore, this can be applied to stakeholder-focused approaches to help identify potential attackers at various stages within a system. The problem of focusing only on technical aspects can be addressed by further diversifying solutions to a more social emphasis within socio-technical approach that can be leveraged to create more robust and safe systems.

BIBLIOGRAPHY

[1] M. M. Vai, R. I. Khazan, D. M. Utin, S. R. O'Melia, D. J. Whelihan, and B. R. Nahill, "Secure embedded systems," tech. rep., MIT Lincoln Laboratory Lexington United States, 2016.

[2] Vehicle Cybersecurity Systems Engineering Committee, "J3061 cybersecurity guidebook for cyber-physical vehicle systems," tech. rep., SAE International, 2016.

[3] S. Back and J. LaPrade, "Cyber-situational crime prevention and the breadth of cybercrimes among higher education institutions," vol. 3, pp. 25–47, 2020.

[4] L. E. Cohen and M. Felson, "Social change and crime rate trends: A routine activity approach," *American Sociological Review*, vol. 44, no. 4, pp. 588–608, 1979.

[5] J. Hough, N. Tilley, and G. B. P. R. Group, *Getting the Grease to the Squeak: Research Lessons for Crime Prevention*. Crime detection and prevention series, Home Office Police Research Group, 1998.

[6] E. Monk, Heinonen, "Street robbery guide no. 59." https://popcenter.asu.edu/content/street-robbery-0, May 2010.

[7] R. V. Clarke, "Situational crime prevention," *Crime and Justice*, vol. 19, pp. 91–150, 1995.

[8] B. ROSS, J. RHE, A. M. HILL, M. CHUCHMACH, and A. KATERSKY, "Toyota to pay 1.2b for hiding deadly unintended acceleration," tech. rep., 2014.

[9] M. Wayland, "Gm expects to offer personal self-driving vehicles to consumers this decade," tech. rep., 2021.

[10] O. Avatefipour and H. Malik, "State-of-the-art survey on in-vehicle network communication (can-bus) security and vulnerabilities," *CoRR*, vol. abs/1802.01725, 2018.

[11] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno, *et al.*, "Comprehensive experimental analyses of automotive attack surfaces.," in *USENIX Security Symposium*, San Francisco, 2011.

[12] P. Peris-Lopez, J. C. Hernandez-Castro, J. M. Estevez-Tapiador, and A. Ribagorda, "Rfid systems: A survey on security threats and proposed solutions," in *Personal Wireless Communications*, pp. 159–170, Springer, 2006.

[13] N. Yoshioka, H. Washizaki, and K. Maruyama, "A survey on security patterns," *Progress in informatics*, vol. 5, no. 5, pp. 35–47, 2008.

[14] B. H. C. Cheng, B. Doherty, N. Polanco, and M. Pasco, "Security patterns for connected and automated automotive systems†," *Journal of Automotive Software Engineering*, vol. 1, pp. 51–77, 2020.

[15] B. H. C. Cheng, B. Doherty, N. Polanco, and M. Pasco, “Security patterns for automotive systems,” in *Modeling in Automotive System and Software Engineering Workshop (MASE2019), 2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, pp. 54–63, 2019.

[16] N. Polanco and B. H. Cheng, “Situational crime prevention for automotive cybersecurity,” in *Modeling in Automotive System and Software Engineering Workshop (MASE2022), Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, MODELS ’22, (New York, NY, USA), p. 562–568, Association for Computing Machinery, 2022.

[17] G. Booch, J. Rumbaugh, and I. Jacobson, *The Unified Modeling Language user guide*. USA: Addison Wesley Longman Publishing Co., Inc., 1999.

[18] J. Jürjens, *Secure Systems Development with UML*. SpringerLink, 2005.

[19] N. Polanco and B. H. Cheng, “Socio-technical automotive security design patterns: Applying a stakeholder-based approach to securing self-driving vehicles,” MODELS Companion ’24, (New York, NY, USA), Association for Computing Machinery, 2024.

[20] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*. USA: Addison-Wesley Longman Publishing Co., Inc., 1995.

[21] E. B. Fernandez, N. Yoshioka, and H. Washizaki, “Patterns for security and privacy in cloud ecosystems,” in *Evolving Security and Privacy Requirements Engineering (ESPRE), 2015 IEEE 2nd Workshop on*, pp. 13–18, IEEE, 2015.

[22] Y. Ito, H. Washizaki, M. Yoshizawa, Y. Fukazawa, T. Okubo, H. Kaiya, A. Hazeyama, N. Yoshioka, and E. B. Fernandez, “Systematic mapping of security patterns research,” in *Proceedings of the 22nd Conference on Pattern Languages of Programs*, p. 14, The Hillside Group, 2015.

[23] M. Schumacher, E. Fernandez-Buglioni, D. Hybertson, F. Buschmann, and P. Sommerlad, *Security Patterns: Integrating security and systems engineering*. John Wiley & Sons, 2013.

[24] G. Baxter and I. Sommerville, “Socio-technical systems: From design methods to systems engineering,” *Interacting with Computers*, vol. 23, pp. 4–17, 08 2010.

[25] M. Yar, “The novelty of ‘cybercrime’: An assessment in light of routine activity theory,” *European Journal of Criminology*, vol. 2, no. 4, pp. 407–427, 2005.

[26] E. R. Leukfeldt and M. Yar, “Applying routine activity theory to cybercrime: A theoretical and empirical analysis,” *Deviant Behavior*, vol. 37, no. 3, pp. 263–280, 2016.

[27] T. C. Pratt, K. Holtfreter, and M. D. Reisig, “Routine online activity and internet fraud targeting: Extending the generality of routine activity theory,” *Journal of Research in Crime and Delinquency*, vol. 47, no. 3, pp. 267–296, 2010.

[28] F. M. Company, 2019. Ford Correspondent, SCP-AC Research Group.

[29] A. Greenberg, “Watch this wireless hack pop a cars locks in minutes,” tech. rep., Wired Magazine, apr 2014.

[30] C. P. Pfleeger and S. L. Pfleeger, *Security in Computing (4th Edition)*. USA: Prentice Hall PTR, 2006.

[31] A. O.-. diagnostic scanner online homepage, jan 2012.

[32] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, and Kantor, “Experimental security analysis of a modern automobile,” in *Security and Privacy (SP), 2010 IEEE Symposium on*, pp. 447–462, IEEE, 2010.

[33] C. Miller and C. Valasek, “Adventures in automotive networks and control units,” in *DEF CON 21 Hacking Conference. Las Vegas, NV: DEF CON*, 2013.

[34] B. Amazon, “Buke b16 super mini wireless wifi obd-2 diagnostic scanner.” Advertised online, jul 2015.

[35] K. H. Johansson, M. Törngren, and L. Nielsen, “Vehicle applications of controller area network,” in *Handbook of networked and embedded control systems*, pp. 741–765, Springer, 2005.

[36] O. Avatefipour and H. Malik, “State-of-the-art survey on in-vehicle network communication “can-bus” security and vulnerabilities,” *International Journal of Computer Science and Network*, vol. 6, pp. 720–727, 2017.

[37] J. Kaiser and M. Mock, “Implementing the real-time publisher/subscriber model on the controller area network (can),” in *Object-Oriented Real-Time Distributed Computing, 1999.(ISORC’99) Proceedings. 2nd IEEE International Symposium on*, pp. 172–181, IEEE, 1999.

[38] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno, *et al.*, “Comprehensive experimental analyses of automotive attack surfaces.,” in *Proceedings of the 20th USENIX Conference on Security*, SEC’11, 2011.

[39] D. K. Nilsson, U. E. Larson, F. Picasso, and E. Jonsson, “A first simulation of attacks in the automotive network communications protocol flexray,” in *Proceedings of the International Workshop on Computational Intelligence in Security for Information Systems CISIS’08*, pp. 84–91, Springer, 2009.

[40] I. Rouf, R. Miller, H. Mustafa, T. Taylor, S. Oh, W. Xu, M. Gruteser, W. Trappe, and I. Seskar, “Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study,” in *Proceedings of the 19th USENIX Conference on Security*, USENIX Security’10, (USA), p. 21, USENIX Association, 2010.

[41] T. Hoppe, S. Kiltz, and J. Dittmann, “Security threats to automotive can networks–practical examples and selected short-term countermeasures,” in *Computer Safety, Reliability, and Security*, pp. 235–248, Springer, 2008.

[42] R. Lever, “Hackers can get into most ’connected cars’,” *Yahoo Technology*, feb 2015.

[43] R. Nusser and R. M. Pelz, “Bluetooth-based wireless connectivity in an automotive environment,” in *Vehicular Technology Conference, 2000. IEEE-VTS Fall VTC 2000. 52nd*, vol. 4, pp. 1935–1942, IEEE, 2000.

[44] R. Brooks, S. Sander, J. Deng, and J. Taiber, “Automobile security concerns,” *Vehicular Technology Magazine, IEEE*, vol. 4, no. 2, pp. 52–64, 2009.

[45] S. Indesteege, N. Keller, O. Dunkelman, E. Biham, and B. Preneel, “A practical attack on keeloq,” in *Advances in Cryptology–EUROCRYPT 2008*, pp. 1–18, Springer, 2008.

[46] R. M. Ishtiaq Roufa, H. Mustafaa, S. O. Travis Taylora, W. Xua, M. Gruteserb, W. Trappeb, and I. Seskarb, “Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study,” in *19th USENIX Security Symposium, Washington DC*, pp. 11–13, 2010.

[47] D. Newcomb, “Car tech 101: Telematics system basics.” Online, aug 2011.

[48] A. Greenberg, “Hackers remotely kill a jeep on the highway with me in it,” tech. rep., Wired Magazine, jul 2015.

[49] J. Pagliery, “Chryslers can be hacked over the internet,” tech. rep., CNN Money, 2015.

[50] C. Vallance, “Car hack uses digital-radio broadcasts to seize control,” tech. rep., BBC News, jul 2015.

[51] A. MacGregor, “Hackers seize control of car using dab radio signals,” tech. rep., CloserStill Media, jul 2015.

[52] J. Leyden, “Now car hackers can bust in through your motor’s dab radio,” tech. rep., The Register (U.K.), jul 2015.

[53] J. Wang, J. Liu, and N. Kato, “Networking and communications in autonomous driving: A survey,” *IEEE Communications Surveys Tutorials*, vol. 21, no. 2, pp. 1243–1274, 2019.

[54] A. Rasheed, S. Gillani, S. Ajmal, and A. Qayyum, “Vehicular ad hoc network (vanet): A survey, challenges, and applications,” in *Vehicular Ad-Hoc Networks for Smart Cities* (A. Laouiti, A. Qayyum, and M. N. Mohamad Saad, eds.), (Singapore), pp. 39–51, Springer Singapore, 2017.

[55] S. A. Gillani, F. Shahzad, A. Qayyum, and R. Mehmood, “A survey on security in vehicular ad hoc networks,” in *Communication Technologies for Vehicles*, pp. 59–74, Springer, 2013.

[56] M. Jain and R. Saxena, “Vanet: Security attacks, solution and simulation,” in *Proceedings of the Second International Conference on Computational Intelligence and Informatics* (V. Bhateja, J. M. R. Tavares, B. P. Rani, V. K. Prasad, and K. S. Raju, eds.), (Singapore), pp. 457–466, Springer Singapore, 2018.

[57] X. Han and Q. Tan, “Dynamical behavior of computer virus on internet,” *Applied Mathematics and Computation*, vol. 217, no. 6, pp. 2520–2526, 2010.

[58] D. K. Nilsson and U. E. Larson, “A defense-in-depth approach to securing the wireless vehicle infrastructure,” *Journal of Networks*, vol. 4, no. 7, pp. 552–564, 2009.

[59] A. Weimerskirch, “Automotive and industrial data security,” in *Cybersecurity and Cyber-physical Systems Workshop*, 2012.

[60] C. Miller and C. Valasek, “Remote exploitation of an unaltered passenger vehicle,” *Black Hat USA*, vol. 2015, 2015.

[61] S. Parkinson, P. Ward, K. Wilson, and J. Miller, “Cyber threats facing autonomous and connected vehicles: future challenges,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 11, pp. 2898–2915, 2017.

[62] M. Zoppelt and R. Tavakoli Kolagari, “What today’s serious cyber attacks on cars tell us: Consequences for automotive security and dependability,” in *Model-Based Safety and Assessment* (Y. Papadopoulos, K. Aslansefat, P. Katsaros, and M. Bozzano, eds.), (Cham), pp. 270–285, Springer International Publishing, 2019.

[63] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, “Design patterns: Abstraction and reuse of object-oriented design,” in *European Conference on Object-Oriented Programming*, pp. 406–431, Springer, 1993.

[64] E. Fernandez-Buglioni, *Security patterns in practice: designing secure architectures using software patterns*. John Wiley & Sons, 2013.

[65] C.-W. Lin, B. Zheng, Q. Zhu, and A. Sangiovanni-Vincentelli, “Security-aware design methodology and optimization for automotive systems,” *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 21, no. 1, p. 18, 2015.

[66] R. Wassermann and B. H. C.. Cheng, “Security patterns,” tech. rep., Department of Computer Science and Engineering, Michigan State University, East Lansing, Michigan 48824, 2003. (A shortened version appeared in the Proceedings of Requirements Engineering “Using Security Patterns to Model and Analyze Security Requirements” (with S. Konrad, L. Campbell, and R. Wassermann), IEEE Workshop on Requirements for High Assurance Systems, (RHAS03), September 2003, Monterey, California.).

[67] G. Booch, J. Rumbaugh, and I. Jacobson, *Unified Modeling Language User Guide, The (2Nd Edition) (Addison-Wesley Object Technology Series)*. Addison-Wesley Professional, 2005.

[68] C. V. Kifor and A. Popescu, “Automotive cybersecurity: A survey on frameworks, standards, and testing and monitoring technologies,” *Sensors*, vol. 24, no. 18, 2024.

[69] F. Sommer, J. Dürrwang, and R. Kriesten, “Survey and classification of automotive security attacks,” *Information*, vol. 10, no. 4, 2019.

[70] National Highway Traffic Safety Administration and others, “Cybersecurity best practices for modern vehicles.” Report No. DOT HS, 2016.

[71] H. Booth, D. Rike, and G. Witte, "Nvd."

[72] Microsoft. https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20)redirectedfrom=MSDN.

[73] J. Viega and G. McGraw, *Building Secure Software: How to avoid security problems the right way*. Addison-Wesley, 2002.

[74] D. M. Kienzle, M. C. Elder, D. Tyree, and J. Edwards-Hewitt, "Security patterns repository version 1.0," *DARPA, Washington DC*, 2002.

[75] E. B. Fernandez, H. Washizaki, and N. Yoshioka, "Abstract security patterns," in *Proceedings of the 15th Conference on Pattern Languages of Programs*, p. 4, ACM, 2008.

[76] E. B. Fernandez, N. Yoshioka, H. Washizaki, and J. W. Yoder, "Abstract security patterns for requirements specification and analysis of secure systems.," in *WER*, 2014.

[77] A. V. Uzunov, E. B. Fernandez, and K. Falkner, "Securing distributed systems using patterns: A survey," *Computers & Security*, vol. 31, no. 5, pp. 681–703, 2012.

[78] S. Konrad, B. H.C.. Cheng, L. A. Campbell, and R. Wassermann, "Using security patterns to model and analyze security requirements," *Requirements Engineering for High Assurance Systems (RHAS'03)*, vol. 11, 2003.

[79] G. Macher, H. Sporer, R. Berlach, E. Armengaud, and C. Kreiner, "Sahara: A security-aware hazard and risk analysis method," in *2015 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 621–624, 2015.

[80] T. Zhang, H. Antunes, and S. Aggarwal, "Defending connected vehicles against malware: Challenges and a solution framework," *IEEE Internet of Things Journal*, vol. 1, pp. 10–21, Feb 2014.

[81] A. Daeinabi and A. G. Rahbar, "Detection of malicious vehicles (dmv) through monitoring in vehicular ad-hoc networks," *Multimedia tools and applications*, vol. 66, no. 2, pp. 325–338, 2013.

[82] A. Rawat, S. Sharma, and R. Sushil, "VANET: Security attacks and its possible solutions," *Journal of Information and Operations Management*, vol. 3, no. 1, p. 301, 2012.

[83] G. Samara and Y. Al-Raba'nah, "Security issues in vehicular ad hoc networks (VANET): a survey," *CoRR*, vol. abs/1712.04263, 2017.

[84] S. Rizvi, J. Willet, D. Perino, S. Marasco, and C. Condo, "A threat to vehicular cyber security and the urgency for correction," *Procedia Computer Science*, vol. 114, pp. 100–105, 2017.

[85] G. Yan, D. Wen, S. Olariu, and M. C. Weigle, "Security challenges in vehicular cloud computing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 1, pp. 284–294, 2013.

[86] Q. Wang and S. Sawhney, "Vecure: A practical security framework to protect the can bus of vehicles," in *Internet of Things (IOT), 2014 International Conference on the*, pp. 13–18, IEEE, 2014.

[87] U. E. Larson, D. K. Nilsson, and E. Jonsson, "An approach to specification-based attack detection for in-vehicle networks," in *2008 IEEE Intelligent Vehicles Symposium*, pp. 220–225, 2008.

[88] K.-T. Cho and K. G. Shin, "Fingerprinting electronic control units for vehicle intrusion detection.," in *USENIX Security Symposium*, pp. 911–927, 2016.

[89] M. Wang, D. Liu, L. Zhu, Y. Xu, and F. Wang, "Lespp: lightweight and efficient strong privacy preserving authentication scheme for secure VANET communication," *Computing*, vol. 98, no. 7, pp. 685–708, 2016.

[90] M. Wolf and T. Gendrullis, "Design, implementation, and evaluation of a vehicular hardware security module," in *International Conference on Information Security and Cryptology*, pp. 302–318, Springer, 2011.

[91] F. Picconi, N. Ravi, M. Gruteser, and L. Iftode, "Probabilistic validation of aggregated data in vehicular ad-hoc networks," in *Proceedings of the 3rd international workshop on Vehicular ad hoc networks*, pp. 76–85, ACM, 2006.

[92] M. Ghosh, A. Varghese, A. A. Kherani, and A. Gupta, "Distributed misbehavior detection in VANETs," in *Wireless Communications and Networking Conference, 2009. WCNC 2009.*, pp. 1–6, IEEE, 2009.

[93] M. Karthikeyan and M. Venkatesan, "Design of automotive firewall based on thermal analysis," in *Advances in Design and Thermal Systems* (L. Ganippa, R. Karthikeyan, and V. Muralidharan, eds.), (Singapore), pp. 235–241, Springer Singapore, 2021.

[94] M. J. Haber and B. Hibbert, *Privilege Escalation*, pp. 53–68. Berkeley, CA: Apress, 2018.

[95] E. Vasilomanolakis, J. Daubert, D. Boopalan, and M. Mühlhäuser, "Don't steal my drone: Catching attackers with an unmanned aerial vehicle honeypot," in *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, pp. 1–2, 2018.

[96] Holt and R. Team, 2019. Michigan State University.

[97] R. E. Haas and D. P. F. Möller, "Automotive connectivity, cyber attack scenarios and automotive cyber security," in *2017 IEEE International Conference on Electro Information Technology (EIT)*, pp. 635–639, 2017.

[98] K. Kim, J. S. Kim, S. Jeong, J.-H. Park, and H. K. Kim, "Cybersecurity for autonomous vehicles: Review of attacks and defense," *Computers Security*, vol. 103, p. 102150, 2021.

[99] Z. Wang, H. Wei, J. Wang, X. Zeng, and Y. Chang, "Security issues and solutions for connected and autonomous vehicles in a sustainable city: A survey," *Sustainability*, vol. 14, no. 19, 2022.

[100] T. J. H. Jay Kennedy and B. H. Cheng, "Automotive cybersecurity: assessing a new platform for cybercrime and malicious hacking," *Journal of Crime and Justice*, vol. 42, no. 5, pp. 632–645, 2019.

[101] S. Piasecki, L. Urquhart, and P. D. McAuley, "Defence against the dark artefacts: Smart home cybercrimes and cybersecurity standards," *Computer Law and Security Review*, vol. 42, p. 105542, 2021.

[102] C. C. Palmer, "Ethical hacking," *IBM Syst. J.*, vol. 40, p. 769–780, mar 2001.

[103] X. Sun, F. R. Yu, and P. Zhang, "A survey on cyber-security of connected and autonomous vehicles (cavs)," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6240–6259, 2022.

[104] J. Kennedy, 2019. Personal Communication.

[105] B. E. Strom, A. Applebaum, D. P. Miller, K. C. Nickels, A. G. Pennington, and C. B. Thomas, "Mitre attack: Design and philosophy," in *Technical report*, The MITRE Corporation, 2018.

[106] W. Stallings, *Cryptography and Network Security: Principles and Practice*. USA: Prentice Hall Press, 6th ed., 2013.

[107] S. Jeong, M. Ryu, H. Kang, and H. K. Kim, "Infotainment system matters: Understanding the impact and implications of in-vehicle infotainment system hacking with automotive grade linux," in *Proceedings of the Thirteenth ACM Conference on Data and Application Security and Privacy*, CODASPY '23, (New York, NY, USA), p. 201–212, Association for Computing Machinery, 2023.

[108] S. Alyahya, "Crowdsourced software testing: A systematic literature review," *Information and Software Technology*, vol. 127, p. 106363, 2020.

[109] A. Matthew, "Crowd-sourced funding - an innovative funding model for innovative business." April 2016.

[110] D. M. Marinova, "On the use of crowdsourcing labor markets in research," *Perspectives on Politics*, vol. 14, no. 2, p. 422–431, 2016.

[111] B. Schemmann, A. M. Herrmann, M. M. Chappin, and G. J. Heimeriks, "Crowdsourcing ideas: Involving ordinary users in the ideation phase of new product development," *Research Policy*, vol. 45, no. 6, pp. 1145–1154, 2016.

[112] B. Heller, A. Amir, R. Waxman, and Y. Maaravi, "Hack your organizational innovation: literature review and integrative model for running hackathons," vol. 12, 2023.

[113] Y. Li and L. Zhao, "Collaborating with bounty hunters: How to encourage white hat hackers' participation in vulnerability crowdsourcing programs through formal and relational governance," *Information and Management*, vol. 59, no. 4, p. 103648, 2022.

[114] Y. Zhang, S. Egelman, L. Cranor, and J. Hong, "Phinding phish: Evaluating anti-phishing tools," 2007.

[115] H. Shirazi, B. Bezawada, I. Ray, and C. Anderson, “Adversarial sampling attacks against phishing detection,” in *Data and Applications Security and Privacy XXXIII: 33rd Annual IFIP WG 11.3 Conference, DBSec 2019, Charleston, SC, USA, July 15–17, 2019, Proceedings 33*, pp. 83–101, Springer, 2019.

[116] P. Kumaraguru, Y. Rhee, S. Sheng, S. Hasan, A. Acquisti, L. F. Cranor, and J. Hong, “Getting users to pay attention to anti-phishing education: evaluation of retention and transfer,” in *Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit*, pp. 70–81, 2007.

[117] M. L. Hale, R. F. Gamble, and P. Gamble, “Cyberphishing: A game-based platform for phishing awareness testing,” in *2015 48th Hawaii International Conference on System Sciences*, pp. 5260–5269, 2015.

[118] G. Canova, M. Volkamer, C. Bergmann, and R. Borza, “Nophish: An anti-phishing education app,” in *Security and Trust Management* (S. Mauw and C. D. Jensen, eds.), (Cham), pp. 188–192, Springer International Publishing, 2014.

[119] M. E. Moran, “The da vinci robot,” *Journal of Endourology*, vol. 20, no. 12, pp. 986–990, 2006. PMID: 17206888.

[120] R. Keel, “Rational choice and deterrence theory,” *Retrieved July*, vol. 5, p. 2007, 2005.

[121] J. Mirkovic and P. Reiher, “A taxonomy of ddos attack and ddos defense mechanisms,” *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 2, pp. 39–53, 2004.

[122] M. Raya and J.-P. Hubaux, “The security of VANETs,” in *Proceedings of the 2nd ACM international workshop on Vehicular ad hoc networks*, pp. 93–94, ACM, 2005.

[123] T. Zimmermann, R. Premraj, N. Bettenburg, S. Just, A. Schroter, and C. Weiss, “What makes a good bug report?,” *IEEE Transactions on Software Engineering*, vol. 36, no. 5, pp. 618–643, 2010.

[124] M. Finifter, D. Akhawe, and D. Wagner, “An empirical study of vulnerability rewards programs,” in *22nd USENIX Security Symposium (USENIX Security 13)*, (Washington, D.C.), pp. 273–288, USENIX Association, Aug. 2013.

[125] A. T. Chatfield and C. G. Reddick, “Cybersecurity innovation in government: A case study of u.s. pentagon’s vulnerability reward program,” in *Proceedings of the 18th Annual International Conference on Digital Government Research*, dg.o ’17, (New York, NY, USA), p. 64–73, Association for Computing Machinery, 2017.

[126] S. Brookfield and C. Gartner, “The impact of pseudoephedrine regulation at australian pharmacies through project stop: A narrative review,” *Drug and Alcohol Review*, vol. 43, no. 1, pp. 325–342, 2024.

[127] Amazon.

[128] P. H. Lee, B. Fu, W. Cai, J. Chen, Z. Yuan, L. Zhang, and X. Ying, "The effectiveness of an on-line training program for improving knowledge of fire prevention and evacuation of healthcare workers: A randomized controlled trial," *PLOS ONE*, vol. 13, pp. 1–15, 07 2018.

[129] P. Kumaraguru, J. Cranshaw, A. Acquisti, L. Cranor, J. Hong, M. A. Blair, and T. Pham, "School of phish: a real-world evaluation of anti-phishing training," in *Proceedings of the 5th Symposium on Usable Privacy and Security*, SOUPS '09, (New York, NY, USA), Association for Computing Machinery, 2009.

[130] A. Zafft and E. Agu, "Malicious wifi networks: A first look," in *37th Annual IEEE Conference on Local Computer Networks - Workshops*, pp. 1038–1043, 2012.

[131] G. Sagers, B. Hosack, R. Rowley, D. Twitchell, and R. Nagaraj, "Where's the security in wifi? an argument for industry awareness," in *2015 48th Hawaii International Conference on System Sciences*, pp. 5453–5461, 2015.

[132] "Lx forums." https://www.lxforums.com//.

[133] "Jaguar forums." https://www.jaguarforums.com/.

[134] B. R. Jones, "Virtual neighborhood watch: Open source software and community policing against cybercrime," *J. Crim. L. & Criminology*, vol. 97, p. 601, 2006.

[135] K. Kate, S. Negi, and J. Kalagnanam, "Monitoring food safety violation reports from internet forums," in *e-Health–For Continuity of Care*, pp. 1090–1094, IOS Press, 2014.

[136] C. C. Freifeld, J. S. Brownstein, C. M. Menone, W. Bao, R. Filice, T. Kass-Hout, and N. Dasgupta, "Digital drug safety surveillance: monitoring pharmaceutical products in twitter," *Drug safety*, vol. 37, pp. 343–350, 2014.

[137] T. Bennett, K. Holloway, and D. Farrington, "The effectiveness of neighborhood watch," *Campbell Systematic Reviews*, vol. 4, no. 1, pp. 1–46, 2008.

APPENDIX A

AUTOMOTIVE SECURITY DESIGN PATTERNS

This appendix presents the collection of the automotive security patterns, described according to the template from Chapter 4. The patterns can be categorized according to those that address security vulnerabilities due to inward facing communication and those that are outward facing, respectively. Underlined Helvetica refers to pattern names, in-line italics refers to UML diagram elements (participants), and courier font refers to security principles/guidelines. The following patterns are more applicable to inward facing communication: *Authorization, Firewall, Multi-level Security, Tamper Resistant Module*, and these patterns are intended to secure outward facing communication: *Blacklist, DDoS Redundancy, Multi-Factor Authentication, Symmetric Encryption, Third-Party Validation*.

A.1 Authorization

A.1.1 Pattern Name and Classification

The *Authorization* pattern is a Structural Security Pattern.

A.1.2 Intent

Authorization provides a structure that facilitates access control to resources.

A.1.3 Motivation

Many systems need to restrict access to their resources according to certain criteria (e.g. a security policy) [64]. In automotive systems, inadequate authentication and authorization protocols have exposed the system to a variety of security exploits such as attacks on inter-vehicle networks [36], spoofing ECUs [39], and various other exploits [38, 60] . In the case of Cranchelli et al. [1], the motivation for developing an authentication protocol for a communications bus, similar to a CAN bus, comes from the potential for ECUs and ECU messages to be spoofed. According to SAE standard J3061, **preventing unauthorized access to data** as well as controlling component privileges are key principles in developing automotive cybersecurity for a system [2]. According to Viega McGraw [73], the principle of being **reluctant to trust** improves the systems abilities to mitigate any security exploits a sub-system may have. This is particularly applicable to an automotive system, as many components come from different principle, and increasingly, vehicles have more communication with external entities.

A.1.4 Properties

The *Authorization* can be used to satisfy the Authentication property, and the Authorization property.

A.1.5 Applicability

Authorization is applicable to attack Prevention.

A.1.6 Structure

The *Authorization* structure of a system can be captured in terms of their relationships (Figure A.1). Active entities are represented by instances of a *Subject* class and passive resources by instances of a *Protection Object* class. Between those main participants exists a relation that portrays which *Subject* is entitled to access certain objects. The properties of this relationship are organized in the association class *Rights*. The objects of this class define the type of access, transfer conditions, and constraints that restrict the use of *Rights*.

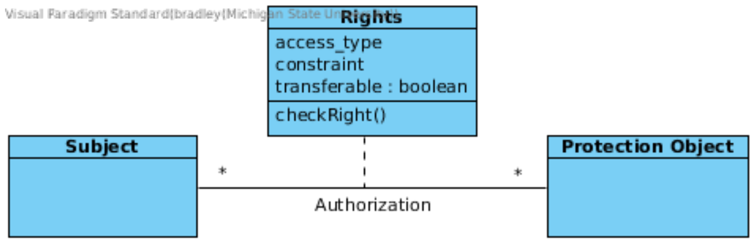


Figure A.1 Class Diagram for Authentication Pattern

A.1.7 Participants

The following section describes the participating classes in the pattern structure.

- *Protection Object*
  - Represents passive resources of the system that are accessed by *Subjects*.
- *Rights*

  - Defines the properties of the authorization rule between *Subject* and *Protection Object*.
  - The access type property describes which kind of access of the current *Rights* object grants. Commonly, this property holds values in the style of read, write, execute, ...
  - The constraint property is a predicate that describes under which circumstances the current *Rights* object is valid and may grant a certain privilege.
  - The transferable property determines whether a right is transitive and its connected *Subject* may grant the right to other active entities.

- *Subject*
  - Represents active entities that need to access *Protected Objects* in accordance to their *Rights*.

A.1.8 Collaborations

Different *Subjects* in the system want to access *Protection Objects*. In order to make use of a certain resource they need to request access to it from the responsible controlling instance. This instance will check if an association class between the *Subject* and the *Protected Object* exists that justifies the required access request. Depending on this examination, access is granted or not.

A.1.9 Behavior

A *Subject* wishing to access a *Protected Object* will request access at a *Checkpoint Object* (Figure A.2). The *Checkpoint* will request the rights pertaining to the *Subject*, and if approved, will forward the request to the *Protected Object.*

A.1.10 Constraints

In an automotive system, example constraints on an *Authorization* pattern include the performance of the authorization protocol and the amount of resources it may require. Because a vehicle is limited in the resources it has available (e.g.. CPU and memory), there is a need to perform authorization efficiently. In addition, the need for real time execution; where multiple *Subjects* may be sharing access to a *Protected Object*, delay is often unacceptable.

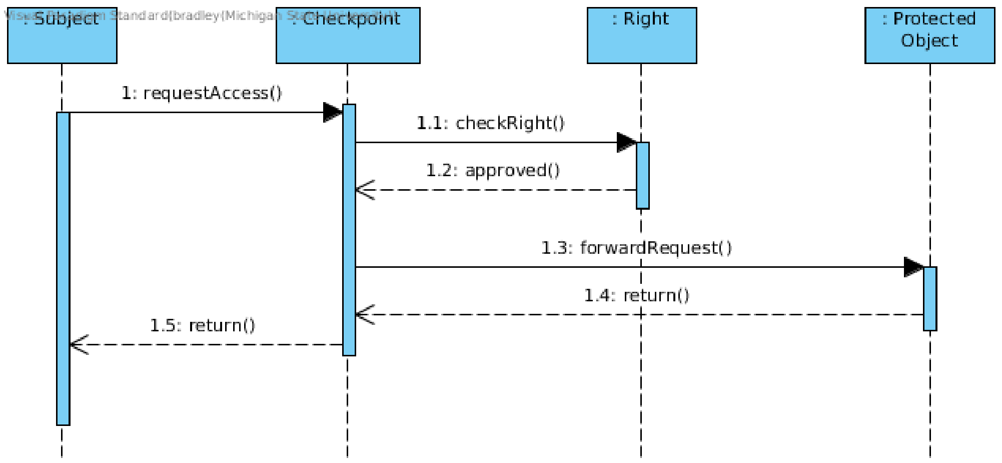


Figure A.2 Sequence Diagram for Authentication Pattern

A.1.11 Consequences

Table A.1 describes consequences in implementing the pattern.

Table A.1 Consequences for Authorization Pattern

Accountability:	Not affected.
Confidentiality:	Can be improved by specification and rigorous enforcement of rights.
Integrity:	Can be improved by specification and rigorous enforcement of rights.
Availability:	Can be improved by specification and rigorous enforcement of rights.
Performance:	The system performance might be derogated by extensive right checks and the evaluation of the rights constraint predicates. This issue is of particular concern in a real-time environment.
Cost:	Cost is not significantly affected by the application of this pattern. May require additional hardware to perform authentication.
Manageability:	Modify to account for automotive systems e.g., ECUs, communication privileges, etc. instead of traditional computing terminology.
Usability:	If the access rights are checked extensively the user might recognize a loss of performance. Authorization may limit utilization of shared resources, impede safety-critical sub-systems.

A.1.12 Known Uses

As mentioned in the motivation section, Cranchelli [1] has developed an authorization module for a shared communication bus to authenticate and manage authorized communication between ECUs on the bus [1]. It is able to authenticate ECUs on the bus at system start-up, and by using digital signatures, authorize ECUs' interactions with one another.

A.1.13 Related Security Patterns

The *Check Point* pattern can be used to examine requests by using the structure of the *Authorization* pattern. The *RBAC* or Roles pattern is an extended version of this pattern.

A.1.14 Supported Principles

The *Authorization* pattern should be used in combination with Principle 4 (**principle of least privilege**) to assign the user's rights. Furthermore, *Authorization* can be used to **compartmentalize** (Principle 5) the system and reduce the impact of a security breach. For example, an attacker that illegally manages to authenticate as user A, has only the rights user A would have. Given a restrictive security policy, the enforcement of access rights **promotes privacy** (Principle 7).

A.2 Firewall

A.2.1 Pattern Name and Classification

Firewall is a structural security pattern.

A.2.2 Intent

A firewall allows for network traffic to be filtered by a set of predefined rules to prevent malicious intrusion.

A.2.3 Motivation

Application firewalls allow for control protocols to be placed on traffic accessing specific services in a subsystem. This may be appropriate when system level protocols are not able to capture the requirements of application level control [64]. In an automotive network, unfiltered traffic can result in attacks on system components from malicious traffic, as well as from compromised system components. On a CAN-Bus for example, externally facing ECUs such as a media player can be compromised and as a result present a threat to other ECUs on the CAN-Bus. Rizvi et al. [84] specifically addressed this scenario in their work. As it pertains to SAE J3061, the use of *Firewall* promotes the principle of **practicing defense in depth**, by securing ECUs on an individual basis [2]. This pattern also promotes the Viega and Mcgraw principles of **Reluctance to Trust**, and **Compartmentalization** [73].

A.2.4 Properties

The *Firewall* can be used to satisfy the Authentication property, the Authorization property, the Integrity property, and the Non-Repudiation property.

A.2.5 Applicability

The pattern is applicable to attack Prevention and attack Detection.

A.2.6 Structure

A *Client* accesses a *Service* through a *Firewall* (Figure A.3). The *Firewall* references the rules which are represented by a *Policy Object* and aggregated in the *Policy Base Object*. The *Firewall* consists of the *Policy Authorization Point*, where the rules and identities for the *Firewall* are centrally stored. The *Firewall* also consists of several *Policy Enforcement Points* which check accesses to a particular module. The data flowing through the *Firewall* is checked by the *Content Inspector*.

A.2.7 Participants

- *Client*
  - Actor requesting access through the *Firewall*.
- *Application*
  - Module protected by *Firewall*, composed of *Services*.
- *Firewall*
  - Enforcement and access point through which the *Client* accesses an *Application*'s *Services*.
- *Policy Authorization Point*
  - Central collection of access policies and identities used by the *Firewall*.
- *Identity Base*
  - Collection of authorized *Identities*.

- *Policy Base*
  - Collection of authorized *Policies*.
- *Policy Enforcement Point*
  - Checks access to *Applications*
- *Content Inspector*
  - Checks data flowing through *Firewall*

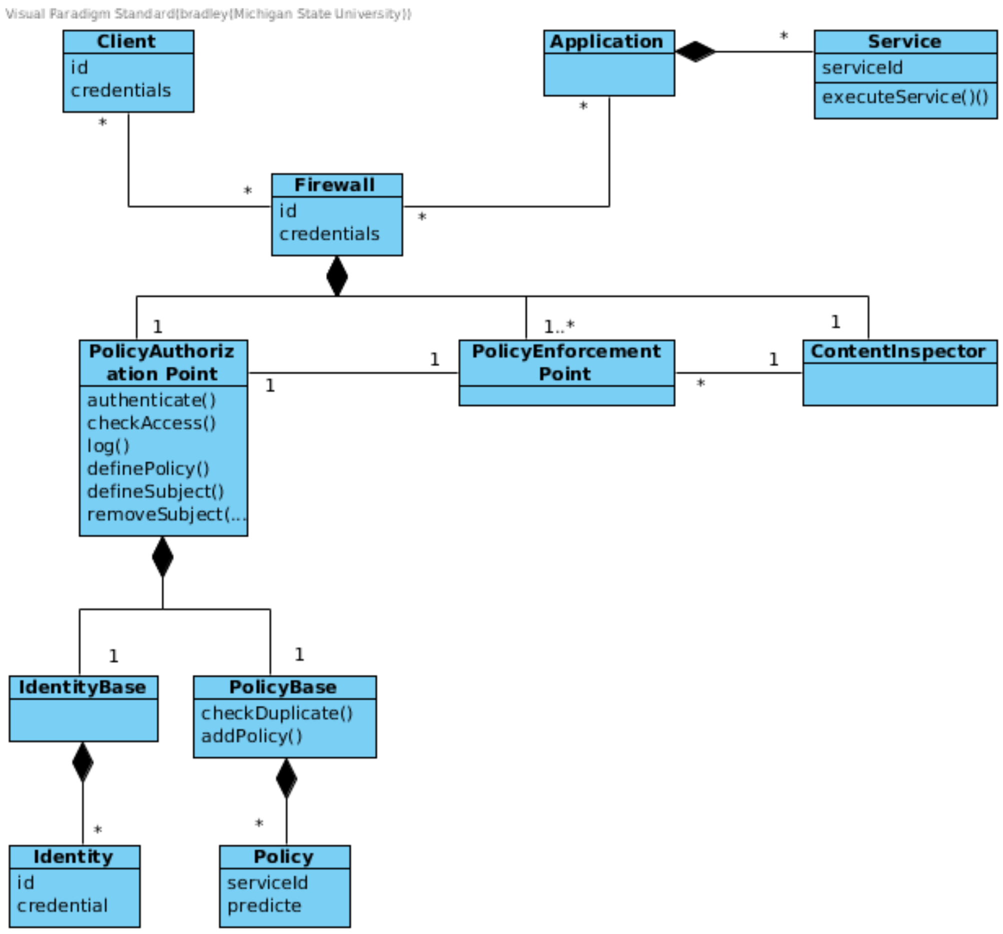


Figure A.3 Class Diagram for Firewall Pattern

A.2.8 Collaborations

A *Firewall* can act as a mediation point for many *Applications* and many *Clients*. An *Application* is composed of *Services*. A *Firewall* is composed of a *Policy Authorization Point*, possibly several *Policy Enforcement Points*, and a *Content Inspector*. The *Policy Authorization Point* is composed of an *Identity Base* and a *Policy Base* which are respectively composed of *Identities* and *Policies*.

A.2.9 Behavior

A *Client* begins by requesting a *Service* from an *Application* through the *Firewall* (Figure A.4). The *Firewall* forwards the request to the *Policy Enforcement Point* where a *Policy Authorization Point* is asked to check the *Client*'s access privileges by looking up the *Identity* and *Policies* in the *Identity Base* and *Policy Base* respectively. If access is granted, then content of the request is checked by the *Content Inspector* and finally sent to the *Application* to service the request.

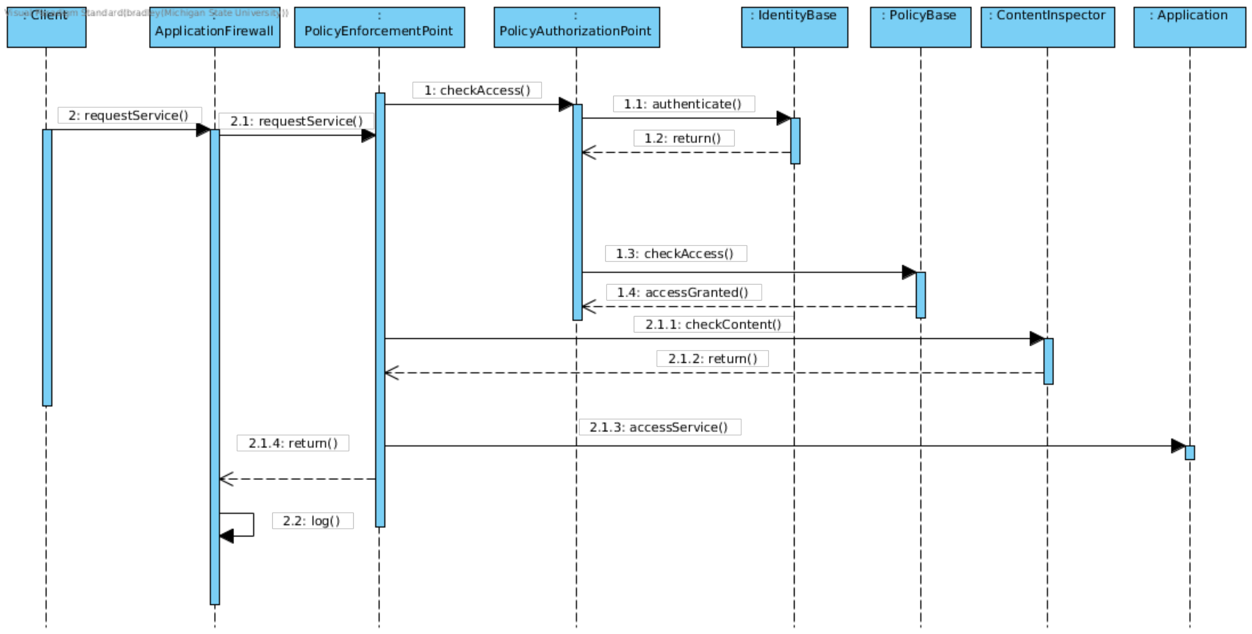


Figure A.4 Sequence Diagram for Firewall Pattern

A.2.10 Constraints

In an automotive system, limited and shared resources limit the extent of *Firewall* usage. Given the constraints of a real-time environment, overhead incurred by processing messages may affect critical response time. Finally, given a variety of different ECU vendors, there can be difficulty in developing a uniform *Firewall* system and enforcing the practice.

A.2.11 Consequences

See Table A.2.

Table A.2 Consequences of Firewall Pattern

Accountability:	Accountability is not affected.
Confidentiality:	Confidentiality can be better protected by controlling access to a particular resource.
Integrity:	Integrity of the system is better maintained by enforcing defense on an individual-module basis.
Performance:	Performance of the system may be affected by overhead of validating senders through the *Firewall*.
Cost:	May require additional hardware.
Manageability:	May require additional management overhead for managing individual ECU compliance and uniformity of a protocol across vendors.
Usability:	Usability of system resources may be affected by overhead of the *Firewall*.

A.2.12 Known Uses

As described previously, a solution to systemic risk of attack on the CAN network given the compromising of an ECU was proposed by Rizvi et al. Here, firewall like programs installed on individual ECUs examine incoming packets to determine if the message should pass to the ECU for processing [84].

A.2.13 Supported Principles

Firewall pattern promotes Principle 1 (**Secure the Weakest Link**), Principle 4 (**Least Privilege**), and Principle 9 (**Be Reluctant to Trust**).

A.2.14 Related Security Patterns

The pattern is related Authorization and *Role Based Access* [64].

A.3 Multi-Level Security

A.3.1 Pattern Name and Classification

Multi-level Security pattern is a structural pattern.

A.3.2 Intent

This pattern is intended to provide a mechanism for handling access in a system with various security classification levels.

A.3.3 Motivation

In many systems integrity and confidentiality of data need to be guaranteed [64]. In an automotive environment, certain resources are critical to system and user safety (e.g. ABS braking and power steering) and must be given a higher degree of access control. Moreover, on an automotive communication network for example, some ECUs might be externally facing (i.e., communicating with external entities) and are consequently more susceptible to compromises. By segmenting access by degrees of trust in a multilevel security hierarchy, system critical resources can be given this higher degree of access control, and resources more susceptible to compromise can be given limited trust, thus ensuring a higher degree of overall security in the system. In the work by Wang and Sawhney [86], the researchers address the problem of different levels of access control, differentiating between externally facing ECUs and those that manage more system critical components [86]. According to SAE J3061 Standards [2], key cybersecurity principles include protecting sensitive data, using the principle of least privilege, and applying defense in depth. A *Multilevel Security* pattern achieves the aforementioned objectives by securing sensitive resources from less trusted entities, giving individual system components privileges to use other system resources only if they are trusted, and finally ensuring that interactions between different components is authenticated on a case-to-case basis.

A.3.4 Properties

The *Multilevel Security* can be used to satisfy the Authorization property, and the Confidentiality property.

A.3.5 Applicability

The pattern is applicable to attack Prevention, and attack Mitigation.

A.3.6 Structure

To represent the *Multilevel Security* pattern, for each *Subject*, there exists an instance of the *Subject Classification* class and for each *Object* an instance of the *Object Classification* class (Figure A.5). These instances are used to aggregate a *Subject*'s and *Object*'s respective security levels and categories.

A.3.7 Participants

- *Object Category*
  - Defines a category to which an *Object* belongs.
- *Object Classification*
  - Determines the security classification of an *Object* (passive resource that is accessed by *Subjects*).
  - The *Object*'s security classification is represented by a set of *Object Categories* and *Object Levels*.
- *Object Level*
  - Defines the security level of an *Object*.
- *Subject Category*
  - Defines a category to which a *Subject* has access.
- *Subject Classification*
  - Determines the security classification of a *Subject* (active entity that accesses *Objects*).
  - The Subject's security classification is represented by a set of *Subject Categories* and *Subject Levels*.
- *Subject Level*
  - Defines the security level of the *Subject*.

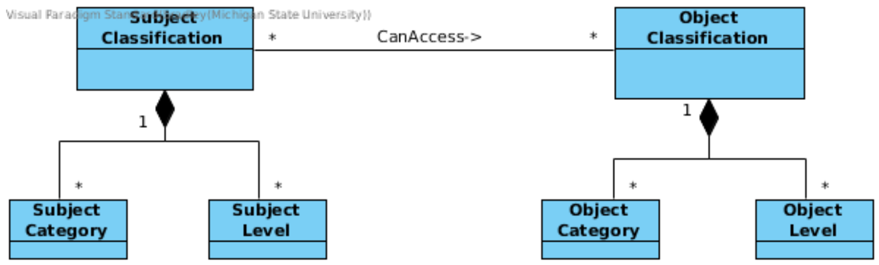


Figure A.5 Class Diagram for Multilevel Security Pattern

A.3.8 Collaborations

The classes *Subject Classification* and *Object Classification* contain a set of *Category* and *Level* classes. This set determines the security classification of the respective *Object*. Access will be granted if the *Classification* of the requesting *Subject* dominates the Classification of the *Object*. The Behavior section explains in detail how to determine whether one classification dominates a second one.

A.3.9 Behavior

In the *Multilevel Security* pattern, the *Subject* requesting access to a resource will request access at the *Checkpoint* (Figure A.6). The *Checkpoint* then obtains the *Classification* of both the *Object* and the requesting *Subject*. If the requesting *Subject*'s classification dominates the *Object*, i.e., has an equal or greater security level, then the request is forwarded.

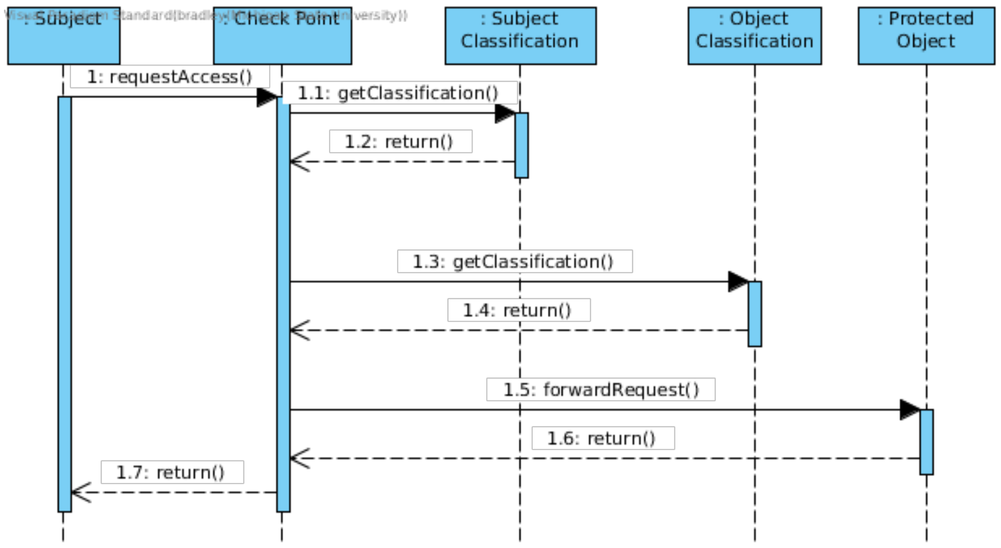


Figure A.6 Sequence Diagram for Multilevel Security Pattern

A.3.10 Constraints

In a real-time environment, such as that of an automotive system, verification of clearance must be a quick and efficient process in order to minimize delays in access to resources.

A.3.11 Consequences

Table A.3 describes the consequences of using the*Multilevel Security* pattern.

Table A.3 Consequences of Multi-Level Security Pattern

Accountability:	Not affected.
Confidentiality:	User access is controlled according to the rules that are stated in the Bell-LaPadula [4] model in order to guarantee confidentiality of data.
Integrity:	Biba's model [28] and its rules establish integrity. Idea of circles of trust in data received.
Availability:	Not affected.
Performance:	The evaluation of access rights can derogate performance.
Cost:	Establishing a multilevel security system can be costly. All subjects and objects need to be classified in certain sensitivity levels. May require additional hardware support.
Manageability:	Not affected.
Usability:	Subjects may be limited in what subsystems they communicate with.

A.3.12 Known Uses

As mentioned previously, the VeCure system seeks to segregate access to certain ECUs by differential degrees of trust. Using multi-tier trust circles, the system is implemented to keep core systems safe from external facing ECUs on a CAN bus network [86]. For example, messages from the externally facing OBD-II port and Telematics systems are considered to be in a lower trust group than the non-externally facing ECUs of the higher trust group, such as the ABS braking system [86].

A.3.13 Related Security Patterns

The *Check Point* pattern may be used to enforce the *Multilevel Security* structure, which can be provided in *Session* object.

A.3.14 Supported Principles

The *Multilevel Security* pattern and its mechanism of granting access to subjects implements the **principle of least privilege** (Principle 4). Furthermore, it facilitates **compartmentalization** (Principle 5) and reduces the impact of a security breach. **Promoting privacy** (Principle 7), **Hiding secrets is hard** (Principle 8), and **Reluctance to trust** (Principle 9) are key ideas of the *Multilevel Security*pattern.

A.4 Signature Based IDS

A.4.1 Pattern Name and Classification

Signature based IDS is a structural based security pattern.

A.4.2 Intent

The pattern provides a mechanism for detecting anomalies in network traffic by using a base-line characteristic of the traffic.

A.4.3 Motivation

On an open network, there is a need for detecting malicious traffic as soon as it occurs to prevent damage to a system. [64] In an automotive network such as that used with the CAN, there is no mechanism for detecting malicious traffic in the protocol, such as a spoofed ECU. By detecting deviations from a baseline behavior, such malicious activity can be detected. SAE J3061 emphasizes the need for comprehensive responses to cybersecurity incidents [2]. A component of this response is the ability to detect intrusions and respond quickly. The pattern supports the principle of Reluctance to Trust [73] as authenticity is always verified before a message is allowed to pass.

A.4.4 Properties

The Signature based IDS can be used to satisfy the Authorization property, and the Integrity property.

A.4.5 Applicability

The Signature based IDS is applicable to attack Detection, attack Prevention, and attack Mitigation.

A.4.6 Structure

The Signature based IDS intercepts the access request for a service (Figure A.7). The *Event Processor* processes the request to parse the relevant *Signature Information*. The *Attack Detector* then tries to match the *Signature Information* against the known signature information to determine if the *Signature* is known. If the *Signature* is unknown, then the appropriate *Response* is issued [64].

A.4.7 Participants

- *IDS:* Intercepts messages and performs proper action given a *Response*
- *Event Processor:* Obtains *Signature* information from message
- *Attack Detector:* Checks *Signature* against known signatures
- *Signature Information:* Has ID of known actor and the corresponding *Signature*
- *Signature:* Identifying characteristic of actor
- *Response:* Communicates signature results back to *IDS* for message handling

A.4.8 Collaborations

The *IDS* has a one-to-one relationship with the *Event Processor* and forwards requests to be processed. The *Event Processor* collaborates in a one-to-one relationship with the *Attack Detector* that obtains processed data from the *Event Processor*. The *Attack Detector* includes the *Signature Information* of known entities. The *Attack Detector* has a one-to-one relationship with the *Response* object that formulates the response given the results of the *Attack Detector*. Finally the *IDS* has a one-to-one relationship with the *Response* object that forwards the correct response to the IDS.

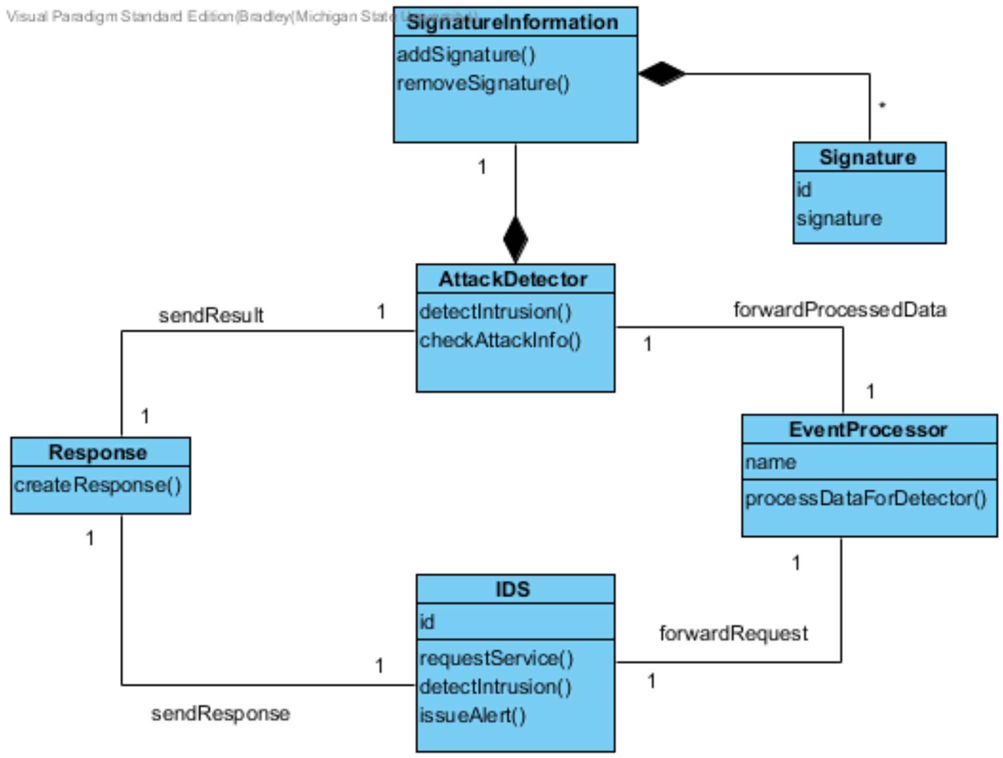


Figure A.7 Class Diagram for Signature IDS Pattern

A.4.9 Behavior

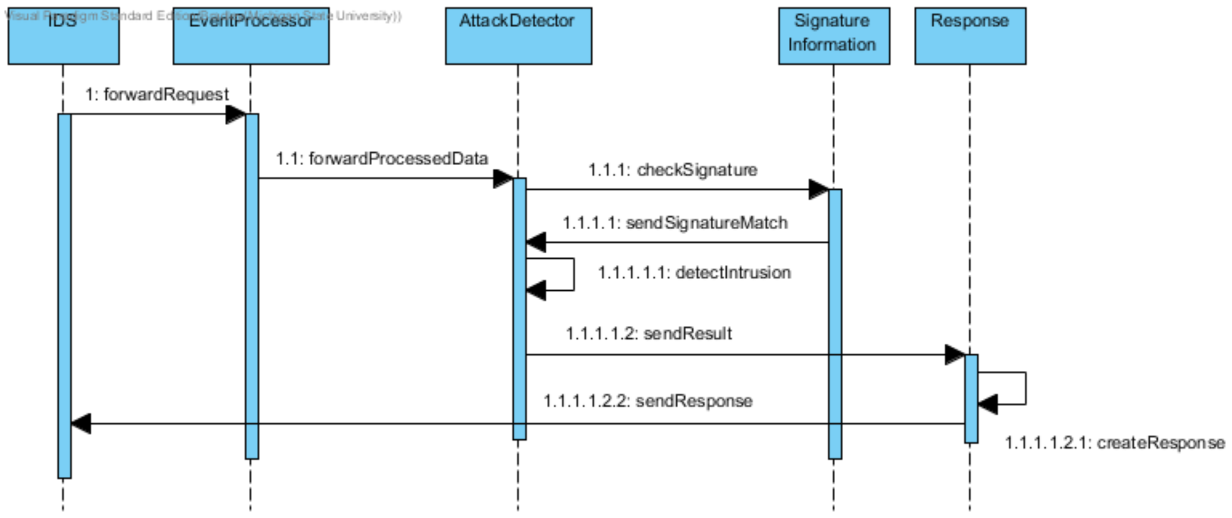
A message is sent from a source to an intended destination node (Figure A.8). The Signature based IDS intercepts the message and determines if the *Signature* is known to the system. If the sender is unknown, then an appropriate *Response* is raised [64].

A.4.10 Constraints

As an IDS analyzes traffic before the messages are sent to the intended receiver, there is overhead incurred when using the pattern. The solution provided by Cho and Shin [88] utilizes a predictive algorithm implemented in hardware to decrease the processing time in order to minimize the overhead.

A.4.11 Consequences

Table A.4 describes the consequences of using the Signature based IDS pattern.



IDS
EventProcessor
AttackDetector
Signature Information
Response
1: forwardRequest
1.1: forwardProcessedData
1.1.1: checkSignature
1.1.1.1: sendSignatureMatch
1.1.1.1.1: detectIntrusion
1.1.1.1.2: sendResult
1.1.1.1.2.2: sendResponse
1.1.1.1.2.1: createResponse

Figure A.8 Sequence Diagram for Signature IDS Pattern

Table A.4 Consequences of Signature Based IDS Pattern

Accountability:	The *Signature based IDS* allows a system to be accountable for the authenticity of the traffic sent on it's network.
Confidentiality:	Not addressed.
Integrity:	By preventing unrecognized agents from communicating on a network, the *Signature based IDS* protects the network from misuse, and the agents attached to the network from attack
Availability:	The *Signature based IDS* may prevent availability on a network as overhead on authentication may lead to under use.
Performance:	The system may take a performance hit in validating the message signatures as it may require significant overhead.
Cost:	Additional hardware to process signatures may incur a cost.
Manageability:	Allows a system to more easily manage the actors that may utilize a network.
Usability:	The usability may decrease as only a small subset of actors may be known to a network.

A.4.12 Known Uses

An example of a lightweight IDS for CAN bus was proposed by Cho and Shin [88] . Here, the system used clock-based finger-printing and predictive algorithms to determine expected clock-skews of ECUs, thus enabling the detection of spoofed messages and authentic ones.

A.4.13 Related Security Patterns

The pattern is a specialization of the Abstract IDS pattern and is related to the *Firewall* pattern, another access control pattern for networks [64].

A.4.14 Supported Principles

Principals supported by the Signature based IDS include Principle 9 (Be Reluctant to Trust), as the IDS intercepts messages to verify the identity of the sender before allowing it to access protected resources, and Principle 3 (Fail Securely).

A.5 Tamper Resistance

A.5.1 Pattern Name and Classification

Tamper Resistantance is a structural based security pattern.

A.5.2 Intent

A *Tamper Resistance*-based module deters unauthorized changing of a system by preventing alterations, or preserving evidence of alteration.

A.5.3 Motivation

In a given system, altering certain modules may lead to unpredictable and potentially dangerous system behavior. In an automotive system, this can result in dangerous vehicle functioning and increased susceptibility to malicious attacks. Work by Wolf and Gendrullis notes the potentially dangerous consequences of alteration to a standard Hardware Security Module [90]. Unchecked alteration of the module can lead to security vulnerabilities for any other vehicle sub-system that relies on the module for cryptography services. As it pertains to SAE J3061, the use of tamper resistant design can promote **the protection of sensitive data**, and **prevention of unauthorized changes to a vehicle** [2]. The pattern also promotes the principles of **failing securely** [73].

A.5.4 Properties

The *Tamper Resistant Modules* can be used to satisfy the Integrity Property, the Non-Repudiation property, and the Authorization property.

A.5.5 Applicability

The pattern is applicable to attack Prevention, attack Mitigation, and attack Detection.

A.5.6 Structure

The *Tamper Resistant Modules* structure of a system can be captured in terms of their relationships (Figure A.9). The pattern is understood as the *Interface*, between the *Subject* and the *Tamper Resistant Object*. The *Interface* has a *Working State* that abstractly describes the untampered status of the *Tamper Resistant Object*. If the *Tamper Resistant Object* is altered, the *Interface* will fail to interact with the *Tamper Resistant Object* in the way the *Working State* predicts, causing the *Interface* to initiate a tamper response, whether it be disabling itself or changing to an immutable tamper state.

A.5.7 Participants

The following section describes the participating classes in the pattern structure.

- *Subject*
  - An actor requesting to access a *Tamper Resistant Object*.
- *Interface*
  - Medium for communication between the *Tamper Resistant Object* and a *Subject*.
- *Working State*
  - Abstractly, the *Interface's* working model of using the *Tamper Resistant Object* as it was designed.

A.5.8 Collaborations

A given *Subject* will attempt to communicate with a *Tamper Resistant Object* through an *Interface*. Consequently, there is an association between both the *Subject* and the *Interface* and the *Tamper Resistant Object* and the *Interface*. The *Interface* has an association with the *Working State*. The *Tamper Resistant Object* has an associated *Working State*.

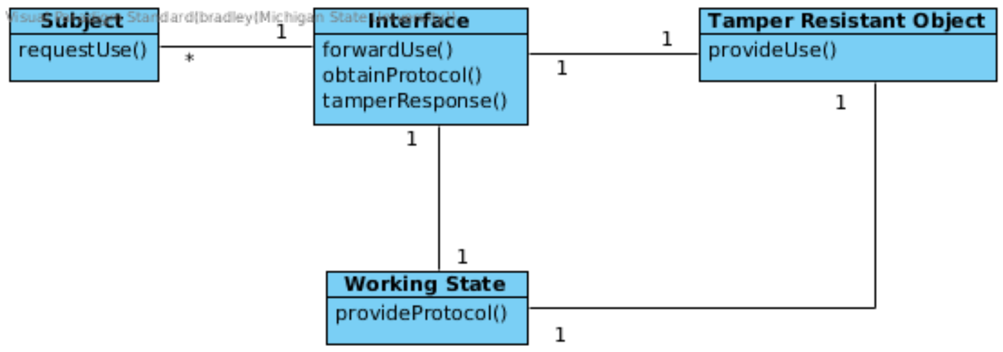


Figure A.9 Class Diagram for Tamper Resistant Modules Pattern

A.5.9 Behavior

Figure A.10 depicts a scenario where a *Subject* interacts with the *Tamper Resistant Object*. The *Subject* will send the request, prompting the *Interface* to forward the request to the *Tamper Resistant Object* on behalf of the *Subject*. To do this, the *Interface* will consult the *Working State* object for the proper protocol of interaction. When the *Interface* fails to interact with the *Tamper Resistant Object* as tampering has rendered the *Tamper Resistant Object* different from its *Working State*, the *Interface* will execute its tamper response.

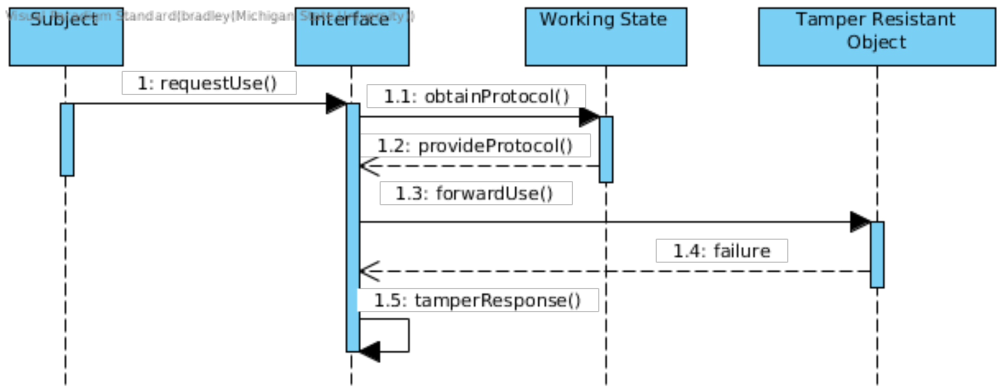


Figure A.10 Sequence Diagram for Tamper Resistant Modules Pattern

A.5.10 Constraints

In an automotive system, additional hardware, like that required to enforce *Tamper Resistant Modules*, can be expensive and take valuable space in the system.

A.5.11 Consequences

Consequences for applying this pattern are given in Table A.5.

Table A.5 Consequences of Tamper-Resistant Modules Pattern

Accountability:	*Tamper Resistant Modules* can increase accountability of a sub-system.
Confidentiality:	N/A
Integrity:	The integrity of a system is improved through the employment of tamper resistance.
Performance:	If implemented in the design of a given module, performance costs should not be a concern.
Cost:	May require additional hardware.
Manageability:	May be more difficult to update a module if additional steps are required to circumvent the tamper protection.
Usability:	Usability should not be affected.

A.5.12 Known Uses

As described by Wolf and Gendrullis [90], a tamper resistant design achieved by a system on chip non-detachable connection with ECU hardware promotes tamper resistance or at least evidence of tampering.

A.5.13 Supported Principles

Tamper Resistant Modules pattern promotes Principle 3 (**Fail Securely**), Principle 4 (**Least Privilege**), J3061 principle 4 (**Prohibit Untested Software Changes**), and J3061 principle 5 (**Prevent Vehicle Owners from Making Unauthorized Changes**) .

A.5.14 Related Security Patterns

Tamper Resistant Modules is related to the *Secure Logger* described by Fernandez [64].

A.6 Blacklist

A.6.1 Pattern Name and Classification

Blacklist is a structural based security pattern.

A.6.2 Intent

A Blacklist intends to monitor the traffic of potentially malicious addresses in a network. Nodes in the network use the Blacklist to block traffic originating from the malicious nodes.

A.6.3 Motivation

Communication with external entities is becoming increasingly integrated into modern automotive systems. Today's cars communicate with smart infrastructure, receive over the air updates from manufacturers, and exchange information with other vehicles on the road. While these technologies allow for improved performance and user experience, the increased reliance on networks leaves a vehicle susceptible to attack from a malicious user on the network. The inability to identify compromised nodes poses a security risk to an automotive system. VANETs allow vehicles and smart infrastructure to share information about current driving conditions within a specific broadcast range. Due to the routing protocols upon which VANETs rely, a malicious node can join a VANET and proceed to attack the network by intercepting the communication between vehicles, dropping packets, and changing or fabricating packets [91] . Applications such as the Post Crash Notification [92] can be deployed on VANETs to detect an attack such as injections of false messages that can lead to potentially dangerous responses by the vehicles on the network.

A.6.4 Properties

The Blacklist can be used to satisfy the Authentication property, the Authorization property, and the Non-Repudiation property.

A.6.5 Applicability

A Blacklist is applicable to the Prevention and Mitigation of an attack.

A.6.6 Structure

The Blacklist structure of a system can be captured in terms of their relationships (Figure A.11). An entity sending a message is captured by a *Client* object. The entity secured with a Blacklist is represented as a *Service* object. The interface between the two objects is a *Checkpoint* object. And finally, the Blacklist captured with the *Blacklist* object.

A.6.7 Participants

The following section describes the participating classes in the pattern structure.

- *Client:* An actor requesting access to a node, or *Service* object.

- *Service:* The intended protected object. A *Service* has access to a system's message processing.
- *Blacklist:* The system's list object for tracking bad addresses.
- *Checkpoint:* The interface through which all communication with a *Service* is achieved, and where the *Blacklist* is checked.

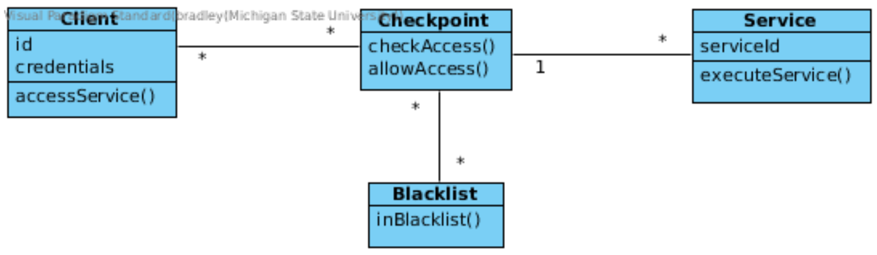


Figure A.11 Class Diagram for Blacklist Pattern

A.6.8 Collaborations

A given *Client* will attempt to communicate with a *Service* through a *Checkpoint*. Consequently, there is an association between both the *Service* and the *Checkpoint* and the *Client* and the *Checkpoint* that acts as the interface. There is also an association between a *Checkpoint* and the associated *Blacklist*.

A.6.9 Behavior

A Blacklist (Figure A.12), is a solution that can partially fulfill the relevant security requirements that are unmet in the problem statement. A Blacklist maintains a list of addresses within a network that have exhibited inappropriate behavior. When a packet from a blacklisted address arrives at a node, the node simply drops the packet. In the context of a VANET, nodes will not process or forward any messages that originate from a blacklisted address, and routing algorithms will not include the blacklisted nodes in calculating routes of the packets.

A.6.10 Constraints

Real-time constraints are important to consider in the context of the Blacklist pattern as message processing incurs overhead as well as consumes additional resources per message.

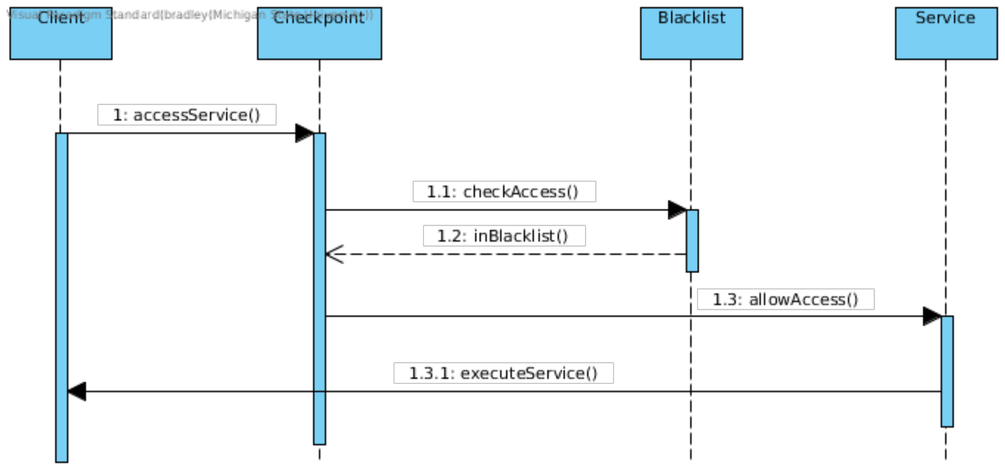


Figure A.12 Sequence Diagram for Blacklist Pattern

A.6.11 Consequences

See Table A.6.

Table A.6 Consequences of Blacklist Pattern

Accountability:	Preventing misbehaving nodes from participating in a network improves the ability of the network to hold malicious actors accountable.
Confidentiality:	Preventing nodes that are known to misbehave from accessing data sent between nodes in the network provides a higher degree of data confidentiality.
Integrity:	Preventing nodes that are known to misbehave from receiving and possibly modifying data sent between nodes in the network provides a higher degree of data integrity.
Availability:	Depending on the blacklisting protocol the Blacklist may prevent harmless nodes from participating in the network, thereby reducing availability.
Performance:	Performance may be affected by the overhead incurred by using the network Blacklist pattern. Performance improvements may occur however as the resources consumed by misuse are reduced .
Cost:	Not applicable.
Manageability:	By setting blacklist rules, manageability of a network can be improved.
Usability:	Depending on the blacklisting protocol, legitimate users may be prevented from accessing the services.

A.6.12 Known Uses

An example of a Blacklist being deployed in an automotive setting is given by Daeinabi and Rahbar [81] . Here the authors propose a system in which a select number of nodes in a VANET are tasked with monitoring behavior of the other nodes, where the overall network can dynamically change over time. If one of the monitoring nodes detects abnormal traffic from a given node, then it increases a distrust value for that node. Meanwhile, all nodes in the network maintain a Blacklist. If a node is given a distrust value beyond a certain threshold by the monitoring nodes, then that node is reported and blacklisted for the other nodes in the network.

A.6.13 Relevant Security Principles

The Blacklist leverages Practice Defense in Depth, Reluctance to Trust, and Compartmentalize.

A.6.14 Related Patterns

The Blacklist is related to traffic filtering patterns such as the Firewall pattern, and the Signature-Based IDS.

A.7 DDoS Redundancy

A.7.1 Pattern Name and Classification

DDoS Redundancy is a structural based security pattern.

A.7.2 Intent

The *DDoS Redundancy* pattern is intended to make a resource or network more resilient to a Distributed Denial of Service attack (DDoS) by providing redundant resources in case a resource becomes inundated with service requests.

A.7.3 Motivation

When a service is provided over a network, resources are typically allotted based on estimated average use. A DoS attack seeks to attack a service's availability to entities that may need to use it by flooding the service with requests. The service will have allocated all of its resources to the attacking node(s) leaving it unavailable to legitimate users. In Vehicular Ad-hoc Networks (VANETs), a DDoS attack may manifest as several nodes broadcasting on a frequency used by the

VANET and consequently jamming the communication medium [82]. The result is the inability for legitimate applications and nodes to send messages on the VANET which may lead to potentially adverse affects.

A.7.4 Properties

The *DDoS Redundancy* can be used to satisfy the Availability property.

A.7.5 Applicability

The *DDoS Redundancy* is applicable to both prevention of an attack, and mitigation of an attack.

A.7.6 Structure

The *DDoS Redundancy* structure of a system can be captured in terms of their relationships (Figure A.13). An entity sending a message is captured by a *Subject* object. The *Subjects* pass their request through a *Check Point* object which manages forwarding the request to the appropriate *Resource* as well as informing the the *Resource Manager* about the request. The *Resource Manager* monitors loads of the system's redundant *Resources* and orchestrates with both the *Check Point* and *Resources* to balance the loads.

A.7.7 Participants

The following section describes the participating classes in the pattern structure.

- *Subject*
  - An actor requesting to access a *Resource*.
- *Check Point*
  - The interface through which all communication with a *Resource* is achieved, and where the *Resource Manager* is updated.
- *Resource*
  - The object requested by the *Subject*.
- *Resource Manager*

– Tracks *Resource* usage and manages loads through orchestration with the *Check Point* and *Resource* objects.

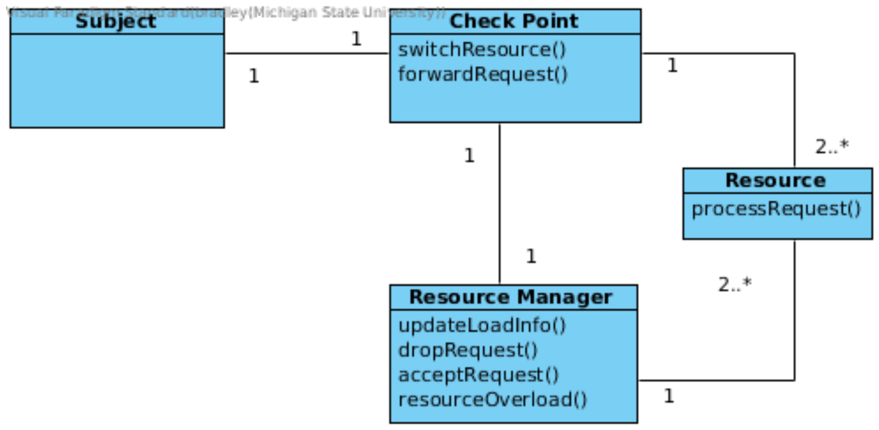


Figure A.13 Class Diagram for DDoS Redundancy Pattern

A.7.8 Collaborations

A given *Subject* will attempt to communicate with a *Resource* through a *Check Point*. Consequently, there is an association between both the *Subject* and the *Check Point* and the *Resource* and the *Check Point*. The *Check Point* has a one to many relationship with *Resources* implying redundant resources. There is also an association between a *Check Point* and the *Resource Manager* which orchestrates forwarding by the *Check Point*. The *Resource Manager* also manages the many *Resources* by tracking loads and sending drop and accept messages based on *Resource* loads.

A.7.9 Behavior

The solution provided by the *DDoS Redundancy* pattern is related to a DDoS defense described by Mirkovic [121] as Resource Multiplication. Many web based service providers deploy this method of redundant web-servers and advanced load-balancers to prevent their web-application from succumbing to a DDoS attack. In the given example of a VANET broadcast frequency being jammed by a DDoS attack, providing several frequencies allows redundant communication mediums for a VANET and consequently allows for a more available service. In Figure A.14, a *Subject* requests a *Resource* at the *Check Point*. The *Check Point* forwards the request to the current

Resource as well as an update to the *Resource Manager*. The *Resource* will not process the packet until it is signaled to do so by the *Resource Manager*. The *Resource Manager* flags the *Resource* as overloaded, and sends a drop request signal to the *Resource* and the resource overload signal to the *Check Point*. The *Check Point* then switches its forwarding protocol to a different *Resource*. The *Subject* will send a request again and this time it will be forwarded to the available *Resource*.

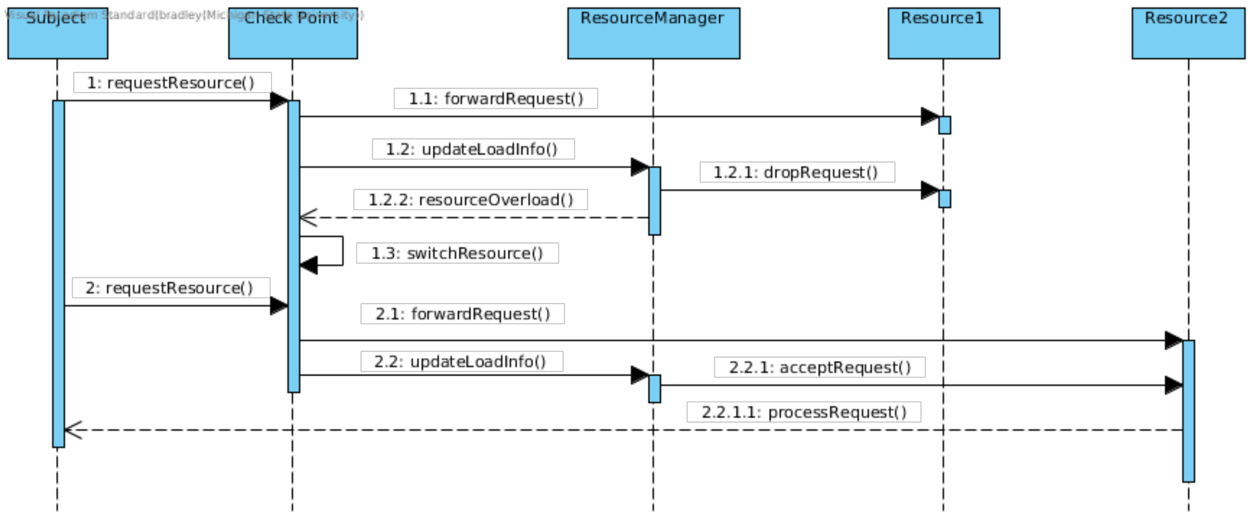


Figure A.14 Sequence Diagram for DDoS Redundancy Pattern

A.7.10 Constraints

Automotive systems are constrained by resource limitations. In the context of *DDoS Redundancy*, this may mean that switching to another resource may not be possible. In a real time environment, overhead required to monitor resources may impede the system response speed while also using more resources per request process.

A.7.11 Consequences

Table A.7 describes how the *DDoS Redundancy* pattern affects the following system characteristics.

A.7.12 Known Uses

While *DDoS Redundancy* is employed widely by web applications today, proposed solutions for DDoS prevention in VANETs are described by Gillani [55]. Researchers proposed switching between communication channels or technologies [122] when one is brought down, while other

Table A.7 Consequences of DDoS Redundancy Pattern

Accountability:	Not affected.
Confidentiality:	Not Affected.
Integrity:	Not Affected.
Availability:	Improves availability by providing redundant resources when a service is inundated with requests.
Performance:	Not affected.
Cost:	The cost of additional resources, such as additional receivers in the previous example, would increase the cost of a system.
Manageability:	Not affected.
Usability:	The use of redundant resources may increase the usability of a given service by providing greater capacity.

researchers [82] propose using multiple receivers operating in disjoint frequencies to provide a feasible solution.

A.7.13 Relevant Security Principles

Relevant security principles related to the *DDoS Redundancy* pattern include Practicing Defense in Depth and Compartmentalization [73] . The automotive security standard SAE J3061 [2] ,promotes defense in depth as a guiding security principle. The *DDoS Redundency* leverages Practice Defense in Depth, Fail Securely, and Compartmentalize.

A.7.14 Related Pattern

The *DDoS Redundancy* makes use of the *Checkpoint* pattern and the *Secure Thread/Process* patterns described by Fernandez [64].

A.8 Multi-Factor Authentication

A.8.1 Pattern Name and Classification

Multi-Factor Authentication is a structural based security pattern.

A.8.2 Intent

The *Multi-Factor Authentication* pattern is used to provide a redundant security measure in authenticating an actor or a message. By enforcing an additional level of authentication, malicious actors can be prevented from attacking if a node or single credential is compromised.

A.8.3 Motivation

While it is common practice to authenticate messages and actors within a system using passwords or signatures, some highly critical systems that house personal data or handle safety critical applications may be susceptible to attack if the passwords or signatures are compromised. If an actor's credentials are compromised, then it is possible to inject false information into a system, steal private data and modify data. In a VANET, if a given node or a vehicle within a smart infrastructure is compromised or an address/signature of the node is replicated (spoofed), the other vehicles in the VANET may be susceptible to a variety of attacks. As an example, in a Sybil attack where a single node sends a message from many fabricated or comprised addresses, applications such as those detecting traffic jams may incorrectly divert other vehicles away from a section of road even though the messages originate from one node [82]. The inability to authenticate these addresses as real vehicles or smart infrastructure could lead to critical safety applications from functioning correctly.

A.8.4 Properties

The *Multi-Factor Authentication* can be used to satisfy the Authentication property, and the Confidentiality property.

A.8.5 Applicability

Multi-Factor Authentication is applicable to the prevention of an attack.

A.8.6 Structure

The *Multi-Factor Authentication* structure of a system can be captured in terms of their relationships (Figure A.15). An entity requesting access is captured by a *Subject* object. The entity secured with a *Multi-Factor Authentication* is represented as a *Protected Object*. The interface between the two objects is a *Checkpoint* object. The *Checkpoint* makes use of multiple *Authenticators* to obtain *Credentials* from a subject.

A.8.7 Participants

The following section describes the participating classes in the pattern structure.

- *Subject*
  - An actor requesting to access to a *Protected Object*.
- *Protected Object*
  - The intended protected object. The entity protected through an authentication *Checkpoint*.
- *Credential*
  - Some identifying secret required to access a *Protected Object*.
- *Checkpoint*
  - The interface through which all communication with a *Service* is achieved, and where the *Blacklist* is checked.
- *Authenticator*
  - Object tasked with retrieving a credential from a *Protected Object*.

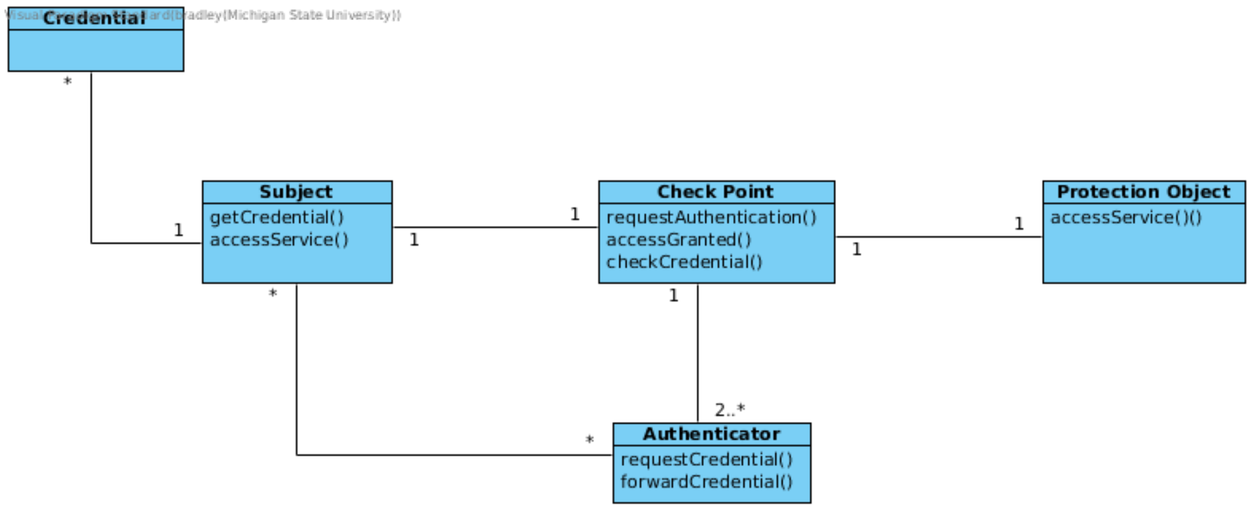


Figure A.15 Class Diagram for Multi-Factor Authentication Pattern

A.8.8 Collaborations

A given *Subject* will attempt to communicate with a *Protected Object* through a *Checkpoint*. Consequently, there is an association between both the *Protected Object* and the *Checkpoint* and the *Subject* and the *Checkpoint* which acts as the interface. There is also an association between a *Checkpoint* and the multiple *Authenticators* associated with it. A given *Authenticator* is associated with a *Subject* which in turn is associated with several *Credentials*.

A.8.9 Behavior

By enforcing the *Multi-Factor Authentication* pattern, a potential attacker is less likely to compromise an actor within a system as obtaining a single credential will not guarantee access to the actor (see Figure A.16. Moreover, other nodes that may communicate with a compromised node are less likely to incorrectly validate spoofed messages given an extra level of authentication. *Multi-Factor Authentication* is used in many web applications where a user is required to enter a password to log in as well as another authentication measure such as a security question the user would know, or a code sent to the user in a separate application such as SMS. In the example given previously, the use of another authenticating measure besides the address of a node would place a greater burden on the attacker using fabricated addresses. The *Multi-Factor Authentication* strengthens an actor's resilience to compromise by strengthening authentication capabilities, and consequently the system's ability to promote integrity and privacy.

A.8.10 Constraints

Automotive systems are limited in the amount of resources that can be provided. By requiring more overhead to authenticate *Subjects*, resources critical to safety may be affected. Furthermore, in VANET applications where an increased amount of nodes participating improves data collection and ability to route messages, adding additional authentication may prove challenging for some harmless nodes resulting in decreased participation.

A.8.11 Consequences

Table A.8 describes how the *Multi-Factor Authentication* pattern affects the following system characteristics.

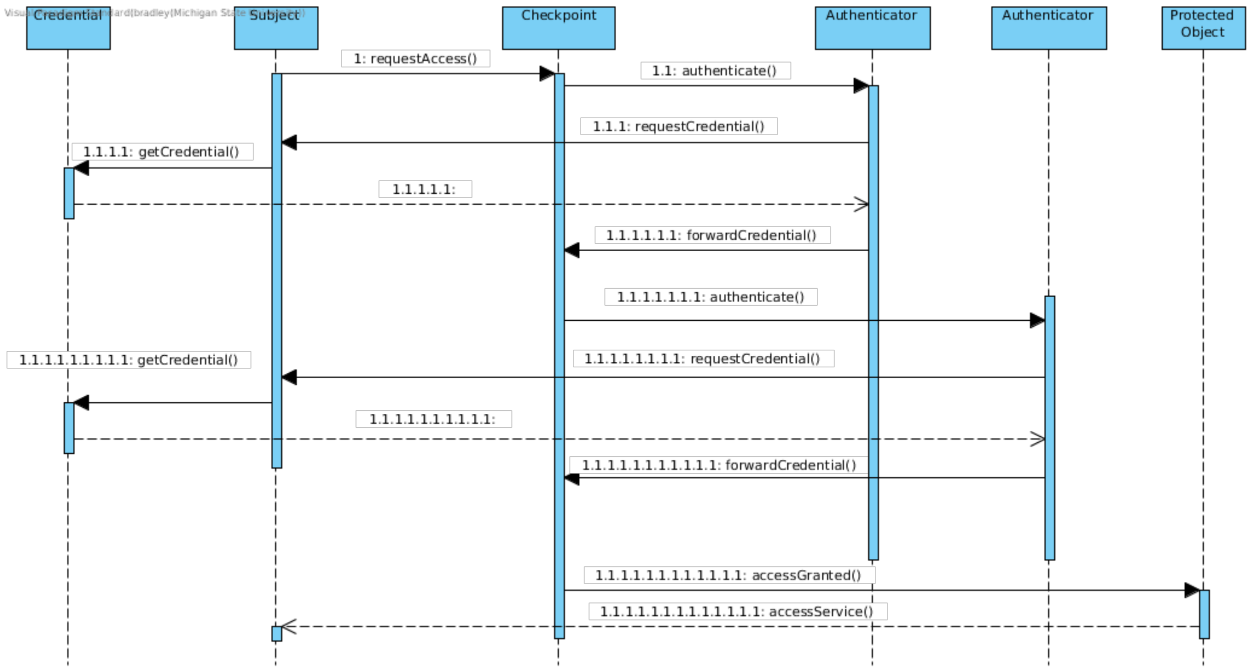

Credential
Subject
Checkpoint
Authenticator
Authenticator
Protected Object
1: requestAccess()
1.1: authenticate()
1.1.1: requestCredential()
1.1.1.1: getCredential()
1.1.1.1.1:
1.1.1.1.1.1: forwardCredential()
1.1.1.1.1.1.1: authenticate()
1.1.1.1.1.1.1.1: requestCredential()
1.1.1.1.1.1.1.1.1: getCredential()
1.1.1.1.1.1.1.1.1.1:
1.1.1.1.1.1.1.1.1.1.1: forwardCredential()
1.1.1.1.1.1.1.1.1.1.1.1: accessGranted()
1.1.1.1.1.1.1.1.1.1.1.1.1: accessService()

Figure A.16 Sequence Diagram for Multi-Factor Authentication Pattern

Table A.8 Consequences of Multi-Factor Authentication Pattern

Accountability:	Accountability is improved as a system is able to more accurately authenticate actors.
Confidentiality:	Private data that may be accessible from compromising an actor is more secured.
Integrity:	The ability to infiltrate and alter data or inject data is made more difficult through more rigorous authentication.
Availability:	Not applicable.
Performance:	Overhead may be incurred by requiring additional authentication steps.
Cost:	The use of other services for additional authentication may incur a cost.
Manageability:	Not applicable.
Usability:	A system may become more difficult to use as multiple steps are required to authenticate legitimate use.

A.8.12 Known Uses

In the given example of the Sybil attack on a VANET, where a single compromised node can fabricate many messages that appear to be from many nodes [82], a proposed solution to detecting the fabrication through the use of multiple points of message authentication has been developed [85]. Here, researchers propose the use of a GPS as a second form of authenticating a message from a given address. If a single node sends many messages with different addresses, then other nodes will be able to see the messages all originating from one node, implying a Sybil attack.

A.8.13 Relevant Security Principles

The *Multi-Factor Authentication* pattern promotes the principles of Practicing Defense in Depth, Compartmentalization, and Being Reluctant to Trust.

A.8.14 Related Patterns

The *Multi-Factor Authentication* pattern is related to the *Authenticator* and can make use of the *Credential* [64].

A.9 Symmetric Encryption

A.9.1 Pattern Name and Classification

Symmetric Encryption is a structural based security pattern.

A.9.2 Intent

Symmetric Encryption is used to encrypt messages so that only a sender and receiver can read the contents.

A.9.3 Motivation

Applications that communicate with external entities may send information that needs to be protected from unintended recipients [64]. In an automotive system that communicates with external entities, such as in a Vehicle Ad-hoc Network (VANET) where a vehicle can be communicating with other vehicles and infrastructure, the inability to accept and process only authentic messages may leave a system susceptible to spoofing and denial of service (DDOS) attacks [38]. In the case of the the LESPP system, researchers seek to mitigate the authentication and privacy preservation challenges facing vehicular ad-hoc networks [89]. As it pertains to SAE J3061 [2], the use of *Symmetric Encryption* aligns with the principles of **protecting personal identifiable information and sensitive data**. The implementation of the pattern also aligns with the Viega and McGraw principles of **promoting privacy** and being **reluctant to trust** [73].

A.9.4 Properties

The *Symmetric Encryption* can be used to satisfy the Confidentiality property.

A.9.5 Applicability

The pattern is applicable to attack Prevention.

A.9.6 Structure

Both the *Sender* and *Receiver* have a *Key* that allows them to decipher a sent *Message* (Figure A.17). A *Sender* will give its *Key* and the *Message* to an *Encryptor* object which will use an *Algorithm* to encrypt the *Message*. The receiver will give its key and the *Encrypted Message* to the *Decryptor* to obtain the original *Message*.

A.9.7 Participants

- *Principal*
  - Main actor that may send and receive messages. Can be defined by either of the inheriting classes *Sender* and *Receiver*.
- *Key*
  - Cryptographic key used by the *Principal* to encrypt and decrypt a message
- *Message*
  - The information to be sent and received. Has inheriting class *Encrypted Message* which is the resulting message after a *Message* undergoes encryption.
- *Encryptor*
  - Uses a *Message* and the *Sender*'s *Key* to create an *Encrypted Message*. Leverages the *Algorithm* Object.
- *Decryptor*
  - Uses an *Encrypted Message* and the *Receiver*'s *Key* to decrypt an *Encrypted Message*. Leverages the *Algorithm* Object.

- *Algorithm*
  - Given a *Message* and a *Key*, either decrypts an *Encrypted Message* or encrypts a *Message*.

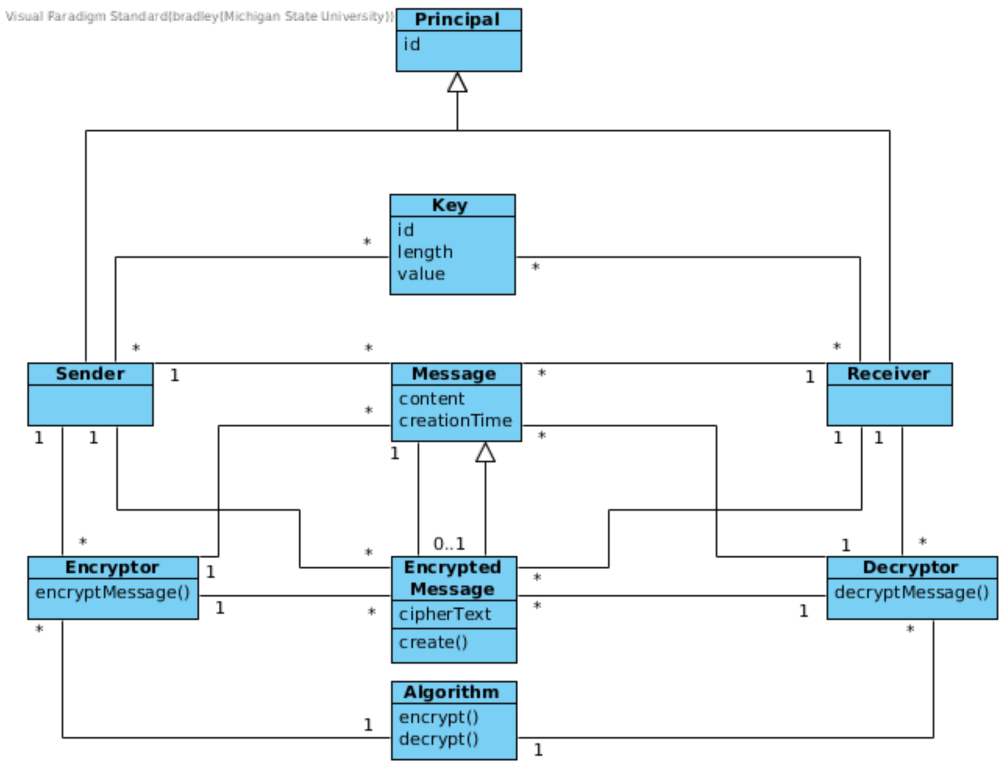


Figure A.17 Class Diagram for Symmetric Encryption Pattern

A.9.8 Collaborations

A *Principal* has a *Key*, *Messages*, and *Encrypted Messages*. The specialized *Principal Sender* has *Encryptors* and the specialized *Principal Receiver* has *Decryptors*. A given *Message* can have up to one associated *Encrypted Messages*. A *Message* has one associated *Decryptor* and one associated *Encryptor*, thus maintaining consistency in the type of encryption and decryption techniques used. Similarly the *Encryptors* and *Decryptors* are associated with a single Algorithm.

A.9.9 Behavior

In the case of sending a *Message*, a *Sender* will send a message and *Key* to a *Encryptor* (Figure A.18). The *Encryptor* will send the information along to the *Algorithm* to encrypt. The

Encryptor can then create the *Encrypted Message* object and return it to the *Sender*.

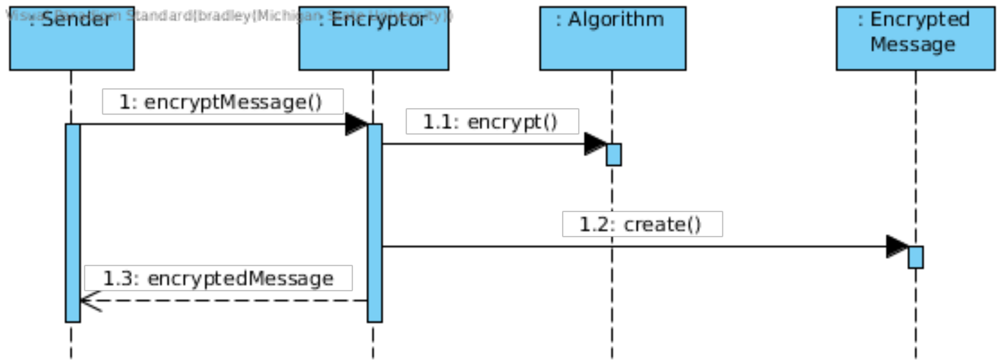


Figure A.18 Sequence Diagram for Symmetric Encryption Pattern

A.9.10 Constraints

In an automotive system where constraints exist on real-time processing and resources, performing the algorithms used in cryptography can be costly in terms of overall system performance. Moreover, overhead is incurred when processing messages to ensure authenticity or to decrypt. In the case of LESPP, computation overhead is reduced by 41.33-77.6% compared to existing public key infrastructures and in simulations, the authors show nearly 0 ms network delay [89].

A.9.11 Consequences

Table A.9 describes the consequences of using the *Symmetric Encryption* pattern.

A.9.12 Known Uses

As described above, the use of symmetric cryptography has been used in the LESPP system described by Wang [89] . By verifying a signature on messages sent over a VANET, LESPP can avoid DDOS and spoofing attacks from unauthenticated nodes.

A.9.13 Supported Principles

Symmetric Encryption supports Principle 7 (**Promoting Privacy**), as well as Principle 9 (**Be Reluctant to Trust**).

A.9.14 Related Security Patterns

The pattern is related to *Asymmetric Encryption* and *Digital Signatures* [64].

Table A.9 Consequences of Symmetric Encryption Pattern

Accountability:	Overall, accountability of a system improves as VANET communication can be authenticated.
Confidentiality:	In the case of LESPP, because the key management center is the only entity that can expose a vehicle's identity, confidential information is better protected.
Integrity:	System information and resources can be better protected from malicious actors.
Performance:	May require additional hardware.
Cost:	Additional hardware to process signatures may incur a cost.
Manageability:	May require additional management overhead for managing certificates of infrastructure and other vehicles.
Usability:	Usability of system resources may be affected by overhead of verifying signatures.

A.10 Third Party Validation

A.10.1 Pattern Name and Classification

Third Party Validation is a structural based security pattern.

A.10.2 Intent

The *Third Party Validation* pattern is intended to provide validation of messages broadcasted in a given network. If a spoofed message is sent to a node and the receiving node cannot validate the message with a trusted node somewhere else in the network, then the receiving node can disregard the message.

A.10.3 Motivation

In many networks that rely on messages being propagated to other nodes, there is concern that a compromised node may alter the message or inject a false message. This type of attack is known as a *Masquerading Attack* that can lead to fabrication, alteration, and replay attacks [55]. In a VANET, many applications, such as an application that reports traffic jams to other vehicles within the VANET, rely on nodes reporting on current conditions as they perceive them and propagating the messages of the nodes around the VANET to create a consensus for the road conditions. If compromised nodes in the VANET send falsified information to another node in the network, the automotive system could behave in an inappropriate manner and jeopardize safety [55].

A.10.4 Properties

The *Third Party Validation* can be used to satisfy the Integrity property, and the Authentication property.

A.10.5 Applicability

Third Party Validation is applicable to Detection and Mitigation.

A.10.6 Structure

The *Third Party Validation* structure of a system can be captured in terms of their relationships (Figure A.19). An entity broadcasting a *Message* is captured by a *Sender* object. Both the *TrustedNode* and *Receiver* actively listen and receive messages on a broadcast network.

A.10.7 Participants

The following section describes the participating classes in the pattern structure.

- *Sender*
  - Broadcasts a given *Message*.
- *Receiver*
  - Listens on a network for a broadcasted *Message*.
- *TrustedNode*
  - Listens on a network for a broadcasted *Message*. Responds to verification requests from *Receivers*
- *Message*
  - Unit of communication over the network.

A.10.8 Collaborations

A given *Sender* broadcasts a *Message* on a network, creating an association with the *Message*. Meanwhile, the other nodes on the network (*Receivers* and *TrustedNodes*) listen for *Messages*,

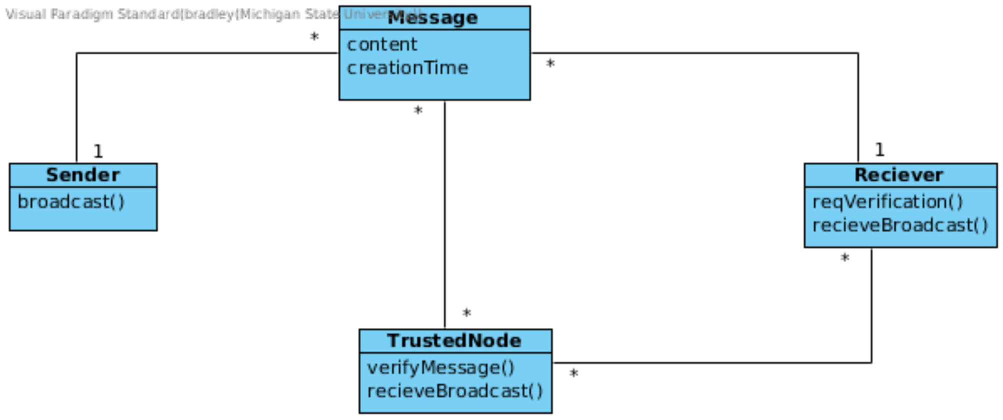


Figure A.19 Class Diagram for Third Party Validation Pattern

creating associations with the *Messages*. Every given *Receiver* node has a one or more *TrustedNodes* used for verification.

A.10.9 Behavior

The *Third Party Validation* pattern seeks to solve the concerns pertaining to integrity by obtaining information from more than one source. Importantly, if there are more trusted nodes within a network that are more secure, they can be leveraged for performing the validation and thus providing a more robust validation scheme. In the case of an application on a VANET, information received by a node can be compared to information received by some other node(s) in the network. If the information is not within some degree of similarity, then the information can be disregarded (see Figure A.20).

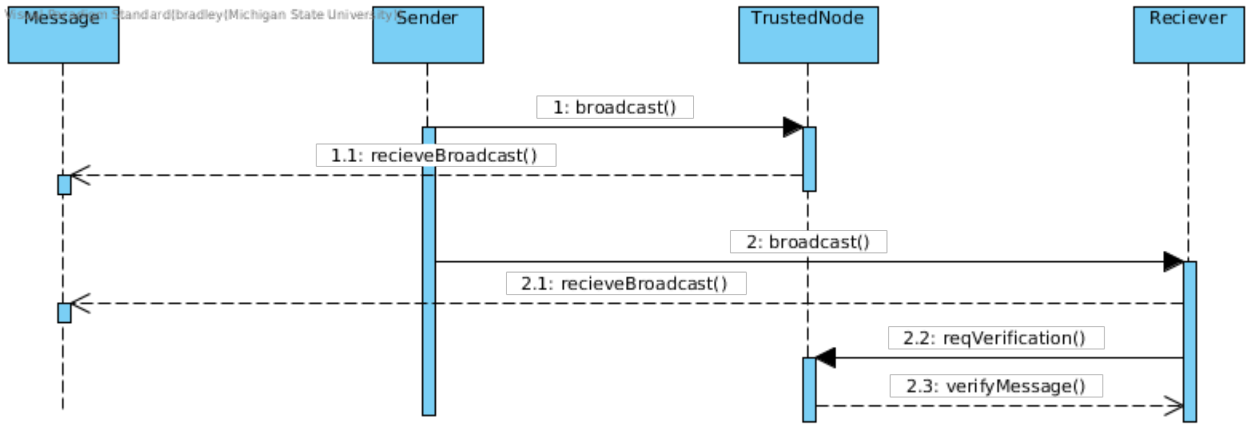


Figure A.20 Sequence Diagram for Third Party Validation Pattern

A.10.10 Constraints

Many automotive systems must function in real time environments to ensure safety. In the context of *Third Party Validation*, adding additional overhead in the verification steps of message processing could add an unacceptable degree of delay.

A.10.11 Consequences

Table A.10 describes how the*Third Party Validation* pattern affects the following system characteristics.

Table A.10 Consequences of Third Party Validation Pattern

Accountability:	Not Applicable.
Confidentiality:	Not Applicable.
Integrity:	Integrity of the system is improved as modified messages are flagged in the validation steps.
Availability:	Not applicable.
Performance:	Overhead from validation may incur performance loss.
Cost:	Not applicable.
Manageability:	Not Applicable.
Usability:	Resources may become more scarce as they are being used for validation.

A.10.12 Known Uses

In a proposed solution by Gillani [55], when a node receives a message in a VANET, the node notifies a Road Side Unit (RSU) and requests to check the message correctness. The RSU will contact the RSU closest to the sending node to validate that the message was actually sent, and not spoofed or modified in transit.

A.10.13 Relevant Security Principles

Principles related to the *Third Party Validation* pattern include Promoting Privacy, and Being Reluctant to Trust.

A.10.14 Related Patterns

This pattern is related to the *Enrolling Using Third-Party Validation* pattern described by [74].

APPENDIX B

SOCIO-TECHNICAL AUTOMOTIVE CYBERSECURITY DESIGN PATTERNS

This appendix presents the collection of the socio-technical design patterns, described according to the template from Chapter 6.

B.1 Bug Exploit Discovery

The Bug Exploit Discovery pattern leverages the number of users on a platform to help identify bugs or flaws through a reporting system that offers additional help to developers searching for vulnerabilities. This can often be accompanied by a monetary reward to dissuade users from exploiting bugs for malicious purposes, and instead contribute to overall security of the system.

B.1.1 Pattern Name and Classification

The Bug Exploit Discovery pattern is a structural security pattern because of policy changes needed by the company. The user interactions cannot be changed, but policies regarding monitoring and regulation need to be applied.

B.1.2 Intent

This pattern provides a structure to encourage and facilitate exploit and bug discovery within the auto-enthusiasts community. Through reward-based engagement, active issues and assistance is given to the company with respect to their software. The goal of creating altruistic users reporting bugs has two main motivations. The first is to decrease the number of exploits found within a system by correcting vulnerabilities that are actively reported by the involved community. The second is to generally increase the number of users who are looking for software flaws. The SCP strategies intended for this pattern can be found in Table B.1.

B.1.3 Also Known As

This pattern also refers to aspects of web-forum mining, community-involved security, and vulnerability reward program.

Table B.1 SCP Strategies for Bug Exploit Discovery

Increase the Effort	**Increase Risks**	**Reduce Rewards**	**Remove Excuses**
Target Hardening	*Entry/Exit Screenings*	*Target Removal*	*Rule Setting*
Access Control	*Formal Surveillance*	*Identifying Property*	Stimulating Conscience
Deflecting Offenders	*Surveillance by Employees*	*Reducing Temptations*	*Controlling Disinhibition*
Controlling Facilitators	*Natural Surveillance*	*Denying Benefits*	*Facilitating* Compliance

B.1.4 Motivation

The problem being addressed by the pattern is to assist in discovering bug exploits surrounding vehicle software. The solution strategy is to encourage enthusiasts in an application area to assist in the security of vehicles, instead of using exploits for malicious purposes. The monetary reward (if applicable) can attempt to dissuade users from exploiting bugs for malicious purposes, and instead incentivize the reporting of issues.

B.1.5 Applicability

Bug Exploit Discovery is applicable to prevention and detection. The applicability can vary based on the software that is being monitored for bugs, depending on user access.

B.1.6 Participants

The following field describes the participating classes in the pattern structure.

UnauthorizedUse: The crime in this instance is the UnauthorizedUse of a bug for nefarious purposes. The User can exploit this to gain access to the asset or a resources.

Functionality: The target is the Functionality of a website, resource, or system. This can have multiple levels of separation, different levels of access control from different users, and may contain bugs. This functionality can range from anything including logging in users or updating website information.

FunctionalityRoutine: The FunctionalityRoutine is the constant actions taken to ensure the system is working properly.

WebsiteOwner: The guardian of the Functionality is also the manager of the Website, with a name and actions including setting rules, securing resources, and watching functionality.

WebsiteOwnerRoutine: The WebsiteOwnerRoutine are the actions taken by the overarching entity, including altering the routine of Functionality or the Website. The actions can also include securing that Functionality, monitoring other users, and making changes to the website.

User: The offender is represented by the User with varying levels of knowledge.

UserRoutine: The UserRoutine is the set of actions taken to exploit a bug within the website (or system). The actions taken by the user may include browsing the website, logging into the account, and finding a bug.

BugExploit: The crime being utilized is BugExploit, where the User finds an issue and exploits it to gain UnauthorizedAccess.

Bug: The Bug is the vulnerability that the User will use to perform their exploit.

Website: This represents the environment where the UnauthorizedUse is taking place. The Website is the environment with a URL, name, and a respective owner.

WebsiteRoutine: The set of actions taken by the Website includes controlling traffic and providing online resources.

RemoveExcuses: This SCP Strategy is RemoveExcuses that is designed with a description of what category type and what strategy type it applies to. The goal would be stimulating conscious, and is achievable by adding a small incentive to bug reporting done by users.

B.1.7 Collaborations

UnauthorizedUse: The UnauthorizedUse is performed by a User using a BugExploit on the Website. This crime is happening to the Functionality of the website.

Functionality : The target of the UnauthorizedUse and is managed by the WebsiteOwner

FunctionalityRoutine: The set of actions taken by the Functionality.

WebsiteOwner: The manager of the Functionality and the Website.

WebsiteOwnerRoutine: The set of actions taken by the WebsiteOwner.

User: The attacker who uses a Bug Exploit to find a Bug and gain UnauthorizedUse.

UserRoutine: The set of actions taken by the User.

BugExploit: The attack used to commit a UnauthorizedUse attack on the Functionality.

Bug: The Bug is the vulnerability that the User will use to perform their BugExploit.

Website: This is where the UnauthoirzedUse crime will occur to the Functionality.

WebsiteRoutine: The set of actions taken by the Website.

RemoveExcuses: The SCP Strategy that is applied to the WebsiteOwner. This can also occur on the User.

B.1.8 Structure

The relevant class and sequence UML diagrams can be found in Figure B.1 for the Bug Exploit Discovery pattern.

B.1.9 Consequences

This following describes consequences in implementing the pattern, see Table B.2. The constraints of this system include only being able to fix issues that are found by enthusiasts. This also relies on their skill and reporting to find vulnerabilities inside the vehicle.

B.1.10 Implementation

This pattern can be executed by partnering with relevant websites, online postings, or offering incentives for assisting and engaging enthusiasts.

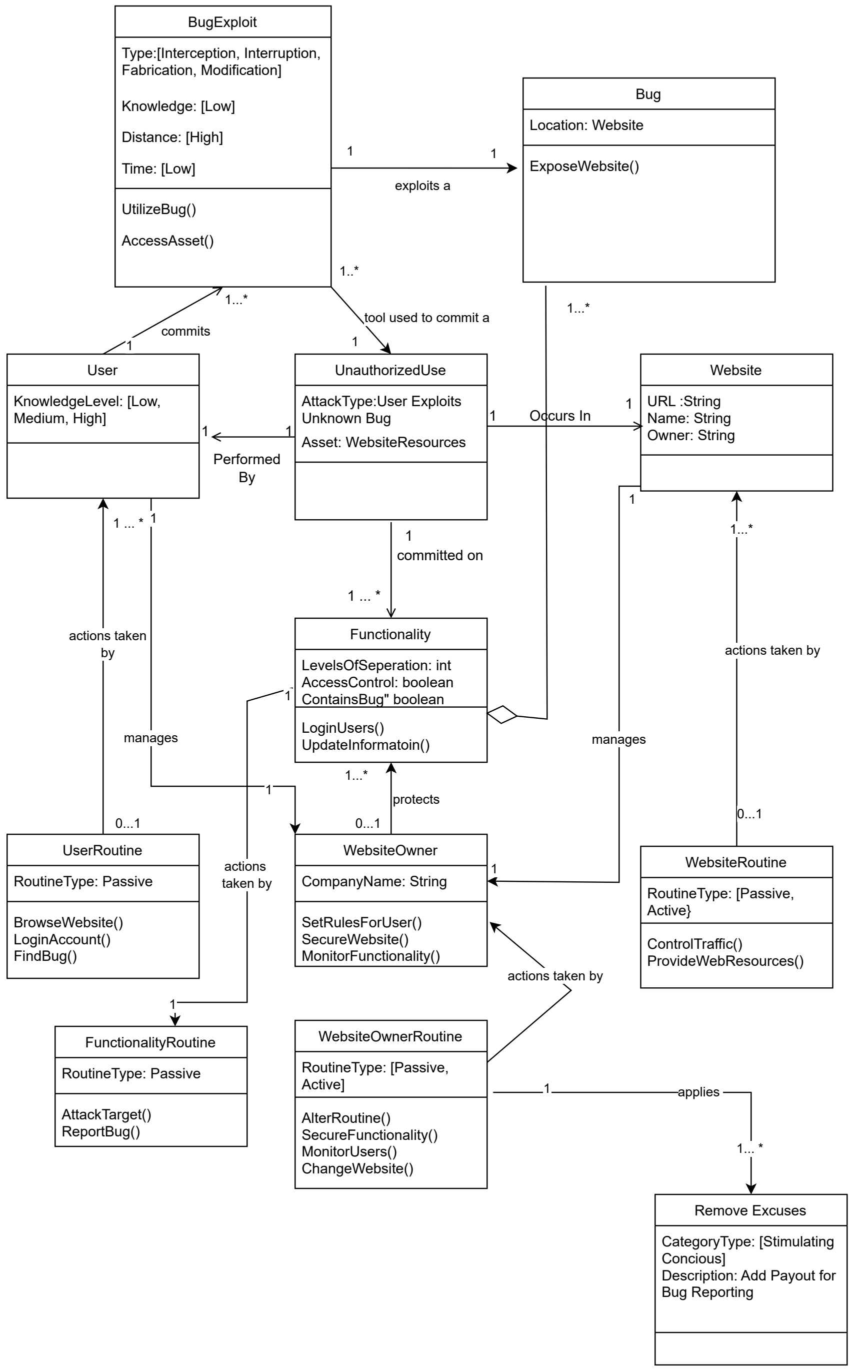


Figure B.1 Bug Exploit Discovery

Table B.2 Consequences for Bug Exploit Discovery.

Accountability:	The User identifying bugs within the system has additional responsibility.
Confidentiality:	N/A
Integrity:	N/A
Availability:	The ability of the User to submit bug-fixes will require infrastructure in place and readily available.
Performance:	The WebsiteOwner needs to be able to promptly address and have a functioning interface.
Cost:	The WebsiteOwner would incur costs for rewarding Users for success.
Manageability:	N/A
Usability:	N/A

B.1.11 Known Uses

The use of tools such as CUEZILLA measures the quality of bug reports and recommends elements that can be added to improve the usability [123]. A number of companies have vulnerability-reward programs (VRP), such as Chrome with over 500 bounties paid and Firefox with around 100. These programs appear economically efficient compared to the cost of hiring full-time security researchers [124]. The U.S Government has also implemented the use of a VRP to assist with cybersecurity measures [125].

B.1.12 Related Patterns

This has a related pattern of Crowdsourcing.

B.2 Tool Restriction

The Tool Restriction pattern uses the relationship with third-party vendors to monitor, control, or restrict the sale of tools that are used to commit attacks on vulnerabilities inside autonomous systems. The limited access to these tools makes executing certain attacks more difficult for the attacker, and helps protect autonomous vehicles.

B.2.1 Pattern Name and Classification

The Tool Restriction pattern is a structural security pattern due to policy changes needed to limit the purchase and acquisition of high-risk tools used for automotive cybersecurity attacks.

B.2.2 Intent

By monitoring the sale of high-risk tools used to cause damage inside the vehicle, potential attacks can be prevented. In the cybersecurity area this is not strictly enforced, where users can purchase the following tools that can be used to conduct a cybersecurity attack: keylogger, RF hacking device, network adapters, RFID scanner, and credit card scanner. This may include legislation and partnership between e-commerce websites about what items should be available for sale. The strategy implemented is by securing the tools necessary to perform various types of attacks, the stakeholders attempting to perform these will have *increased risk*. The specific SCP strategies can be found in Table B.3

Table B.3 SCP Strategies for Tool Restriction

Increase the Effort	**Increase Risks**	**Reduce Rewards**	**Remove Excuses**
Target Hardening	*Entry/Exit Screenings*	*Target Removal*	Rule Setting
Access Control	*Formal Surveillance*	*Identifying Property*	*Stimulating Conscience*
Deflecting Offenders	*Surveillance by Employees*	*Reducing Temptations*	*Controlling Disinhibition*
Controlling Facilitators	*Natural Surveillance*	*Denying Benefits*	Facilitating Compliance

B.2.3 Also Known As

This attack is similar to monitoring purchase of illicit materials in the traditional sense such as allergy medicine (used for drug production), spray paint cans to minors (used for graffiti), or age restrictions on drugs and alcohol (to prevent drug abuse).

B.2.4 Motivation

The Tool Restriction pattern seeks to address the problem of widespread distribution of third-party tools for executing attacks from a variety of stakeholders. The Tool Restriction pattern provides a structure for monitoring the purchase and sale of restricted tools used within autonomous vehicles, reducing the likelihood of attacks being executed and adding risk of discovery by offenders.

B.2.5 Applicability

Tool Restriction is applicable to attack Prevention and Detection. The prevention aspect can be achieved by limiting the sale of illicit tools, and the detection can be done by monitoring who is using or buying restricted tools.

B.2.6 Participants

The following field describes the participating classes in the pattern structure.

InfiltrateSystem: This represents the crime of InfiltrateSystem that requires a tool to commit and target various assets.

SpecifiedComponent: This represents the target of an InfiltrateSystem that focuses on attacking a SpecifiedComponent. The component can perform a given function, and may require a tool to gain access to a SpecifiedComponent.

SpecifiedComponentRoutine: This represents the routine of a SpecifiedComponent that focuses on the actions taken. This will have either an active or a passive routine type, and can do things such as perform functions, monitor access, and enable/disable the system.

Vehicle: This represents the environment where the crime takes place, in this case the Vehicle. The Vehicle may have attributes such as a make, model, and year.

VehicleRoutine: This represents the routine of the Vehicle. The VehicleRoutine can take actions, including securing the vehicle.

Manufacterer: This entity is the caretaker and guardian for both the Vehicle and the Specified-Component. They will have a description, and can take actions to secure the entities it is responsible for.

ManufactererRoutine: The set of actions taken by the Manufacterer. This can be either a passive or active routine, and may try to enhance the security.

MotivatedUser: This represents the offender of a crime as a MotivatedUser with a description of their knowledge level and the ability to purchase a tool necessary to perform an attack.

ToolSeller: This represents the ToolSeller entity responsible for a motivated user and the selling of specified tools. The ToolSeller has a location where the sales take place, whether it has an online platform, and if they can track user sales. They might also have routines such as creating rules, altering items, restricting sales, and tracking users.

ToolBasedAtack: This represents the details of an attack that is a ToolBasedAttack used by the MotivatedUser. The type of attack can be any, will require a medium to high knowledge, and the distance or time will be determined based on the attack. The goal with the attack is to focus on a vulnerability and access the described asset.

ToolRestrictedVulnerbaility: This is the ToolRestrictedVulnerbaility that requires a tool to attack a SpecifiedComponent and perform a given ToolBasedAttack.

RemoveExcuses: This is the SCP strategy of RemoveExcuses that will be applied through rule setting and facilitating compliance. This requires a policy to monitor or restrict sales of high-risk tools.

IncreaseEffort: This is the SCP strategy of IncreaseEffort that will control who can get access to a tool. This can be done by tracking, monitoring, and restricting the usage or purchase of various tools.

B.2.7 Collaborations

This describes the relationships between entities.

InfiltrateSystem: The crime of InfiltrateSystem requires a MotivatedUser to attack a Specified-Component inside the Vehicle. The ToolBasedAttack is the manner in which the Infil-trateSystem is performed.

SpecifiedComponent: This represents the target of an InfiltrateSystem on a SpecifiedComponent. The SpecifiedComponent has a set of actions and is managed by a Manufacterer

SpecifiedComponentRoutine: This represents the set of actions of a SpecifiedComponent.

Vehicle: The environment where the InfiltrateSystem takes place, in this case the Vehicle. The Vehicle has a set of actions and is managed by the Manufacterer.

VehicleRoutine: This represents the set of actions of a Vehicle.

Manufacterer: This is the caretaker and guardian for both the Vehicle and the SpecifiedComponent. The Manufacterer has a set of actions.

ManufactererRoutine: This represents the set of actions of a Manufacterer.

MotivatedUser: The entity committing the InfiltrateSystem using a ToolBasedAttack. The ToolSeller has a level of control over the entity, and does not have a routine as it is untraceable.

ToolSeller: This represents the ToolSeller entity responsible for a MotivatedUser

ToolBasedAtack: The ToolBasedAttack is used by the MotivatedUser to perform the InfiltrateSystem crime. This also exploits a ToolRestrictedVulnerability.

ToolRestrictedVulnerbaility: This is the ToolRestrictedVulnerbaility that requires a tool to attack a SpecifiedComponent and perform a given ToolBasedAttack.

RemoveExcuses: This SCP strategy should be applied to the Manufacterer.

IncreaseEffort: This SCP strategy should be applied to the Manufacterer.

B.2.8 Structure

The relevant class and sequence UML diagrams can be found in Figure B.2 for the Tool Restriction pattern.

B.2.9 Consequences

This following Table B.4 describes consequences in implementing the pattern.

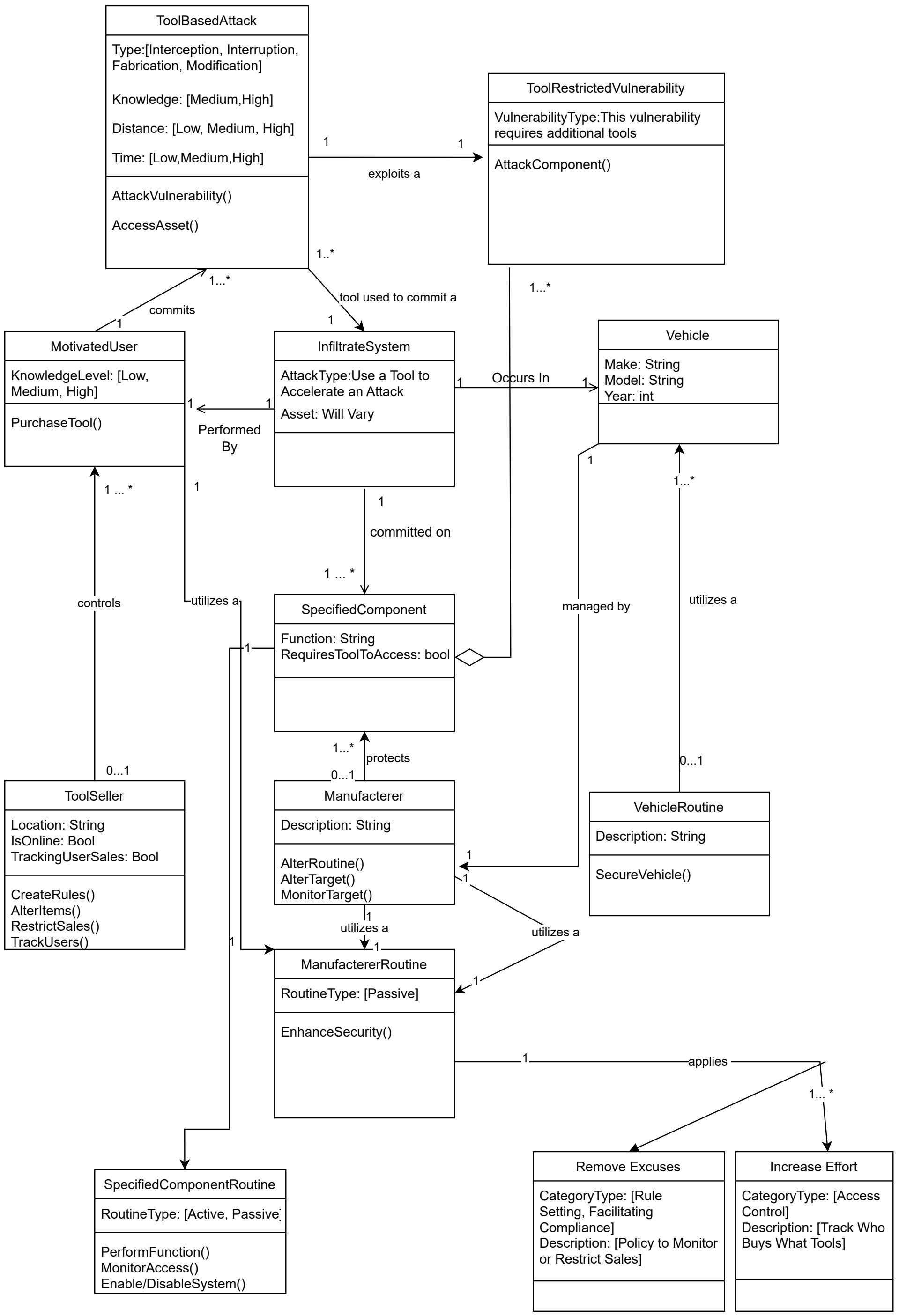


Figure B.2 Tool Restriction

Table B.4 Consequences for Tool Restriction.

Accountability:	The ToolSeller would now be accountable for sales of the item.
Confidentiality:	The MotivatedUser could have information compromised by ToolSeller for buying illegal items.
Integrity:	N/A
Availability:	The items attempting to be purchased would not be easily accessible
Performance:	N/A
Cost:	This would cause ToolSeller sales to go down for certain items, incurring a cost.
Manageability:	The ToolSeller would have more information and users to manage for illicit items.
Usability:	N/A

B.2.10 Implementation

This pattern can be executed by partnering with relevant websites, working with local governments to prevent sales of illicit items, and monitoring purchase of items through undesirable means.

B.2.11 Known Uses

This technique has not been widely applied to address cybersecurity, but has shown success in various traditional crime fields to help prevent crime. For example, Project STOP has been shown as successful in preventing pseudoephedrine from being diverted from pharmacies to methamphetamine production [126]. In the cybersecurity setting, tool purchases for high-risk items such as key-fob monitoring devices, pass-thru devices, and RFID scanners are already widely available [127].

B.2.12 Related Patterns

None

B.3 Gameout

The Gameout pattern seeks to train users on a set of actions to take if they are under the circumstances of an attack while operating an autonomous systems. The set of actions are intended to prevent a cybersecurity threat from advancing, or halt the progress of the attack.

B.3.1 Pattern Name and Classification

The Gameout pattern is a behavioral security pattern that focuses on behaviors that should be executed by stakeholders in order to prevent an attack from escalating.

B.3.2 Intent

The importance of practicing how to handle situations has been present in a traditional sense through fire drills, tornado warnings, and many more. This pattern attempts to take behaviors of practicing necessary procedure prior to an attack in order to educate and streamline efficiency. Using a set of procedures and policies which recommend reporting practices, users would be able to quickly and effectively handle any security breaches regarding autonomous vehicles. Furthermore, by attempting to practice these scenarios users would be better acclimated to an attack when it occurs during operation. The SCP strategies used by this pattern can be found in Table B.5

Table B.5 SCP Strategies for Gameout

Increase the Effort	**Increase Risks**	**Reduce Rewards**	**Remove Excuses**
Target Hardening	*Entry/Exit Screenings*	*Target Removal*	*Rule Setting*
Access Control	*Formal Surveillance*	*Identifying Property*	*Stimulating Conscience*
Deflecting Offenders	*Surveillance by Employees*	*Reducing Temptations*	*Controlling Disinhibition*
Controlling Facilitators	*Natural Surveillance*	*Denying Benefits*	Facilitating Compliance

B.3.3 Also Known As

The Gameout pattern is similar to techniques involved with preventing social engineering attacks in the cyberspace or in a traditional sense similar to fire, tornado, or other drills in schools to better equip staff in an emergency situation.

B.3.4 Motivation

The Gameout pattern assists in lack of education surrounding different cybersecurity attacks for autonomous vehicles by focusing on the stakeholder during an attack. The solution to this problem will encourage companies to practice different scenarios to help prevent attacks, halt an attack, or stop the attacks from escalating to a more severe stage.

B.3.5 Applicability

Gameout is applicable to attack Prevention, Detection, and also Mitigation.

B.3.6 Participants

The following field describes the participating classes in the pattern structure.

SystemTakeover: This represents the crime of SystemTakeover where an attacker is attempting to gain control of a given system. The asset that is being targeted can be specified, but generally the target is control of a subsystem/system.

Driver: This represents the target of a SystemTakeover that focuses on attacking a Driver. The Driver has a level of experience, and an amount of knowledge of technical aspects.

DriverRoutine: This represents the routine of a Driver that focuses on the actions taken. This will have either an active or a passive routine type, and can do things such as drive the vehicle, control system, and enable/disable the system.

Vehicle: This represents the environment where the crime takes place, in this case the Vehicle. The Vehicle may have attributes such as a make, model, and year.

VehicleRoutine: This represents the routine of the Vehicle. The VehicleRoutine can take actions, including securing the vehicle.

SoftwareOrManufacterer: This entity is the caretaker and guardian for both the Vehicle and the Driver. They will have a description, and can take actions to secure the entities it is responsible for.

SoftwareOrManufactererRoutine: The set of actions taken by the SoftwareOrManufacterer. This can be either a passive or active routine, and will try to enhance the security, educate driver, set protocals, and train actions.

Hacker: This represents the offender of a crime as a Hacker with a description of their knowledge level and the ability to purchase a tool necessary to perform an attack.

InfiltrateSystem: This represents the details of an attack that is a InfiltrateSystem used by the Hacker. The type of attack can be any, will require a medium to high knowledge, and distance along with time will be determined based on the attack. The goal with the attack is to focus on a vulnerability and access the described asset.

SystemVulnerbaility: This is the SystemVulnerbaility that attempts to gain control of a system through a security flaw or exploit, but has opportunity where the driver can intervene.

RemoveExcuses: This is the SCP strategy of RemoveExcuses that will be applied through facilitating compliance. This requires a education and regulation of Driver actions to improve the overall safety of the Vehicle.

B.3.7 Collaborations

This describes the relationships between entities.

SystemTakeover: The crime of SystemTakeover requires a Hacker to attack a Driver inside the Vehicle. The InfiltrateSystem is the manner in which the SystemTakeover is performed.

Driver: This represents the target of a SystemTakeover on a Driver. The Driver has a set of actions and is managed by a SoftwareOrManufacterer

DriverRoutine: This represents the set of actions of a Driver.

Vehicle: The environment where the SystemTakeover takes place, in this case the Vehicle. The Vehicle has a set of actions and is managed by the Manufacterer.

VehicleRoutine: This represents the set of actions of a Vehicle.

SoftwareOrManufacterer: This is the caretaker and guardian for both the Vehicle and the Driver. The SoftwareOrManufacterer has a set of actions.

SoftwareOrManufactererRoutine: This represents the set of actions of a SoftwareOrManufacterer.

Hacker: The entity committing the SystemTakehover using a InfiltrateSystem.

InfiltrateSystem: The InfiltrateSystem is used by the Hacker to perform the SystemTakeover crime. This also exploits a ToolRestrictedVulnerability.

SystemVulnerbaility: This is the SystemVulnerbaility that attacks a Target and perform a given InfiltrateSystem.

RemoveExcuses: This SCP strategy should be applied to the SoftwareOrManufacterer.

B.3.8 Structure

The relevant class and sequence UML diagrams can be found in Figure B.3 for the Gameout pattern.

B.3.9 Consequences

This following describes stakeholder focused consequences in implementing the pattern as seen in Table B.6.

Table B.6 Consequences for Gameout.

Accountability:	The Driver has increased responsibility.
Confidentiality:	N/A
Integrity:	N/A
Availability:	The User learning the system would need to use time to learn a process.
Performance:	N/A
Cost:	The SoftwareOrManufacterer would increase costs for the company to compose, train, and practice reporting procedures.
Manageability:	The SoftwareOrManufacterer has increased managing during testing and drills.
Usability:	N/A

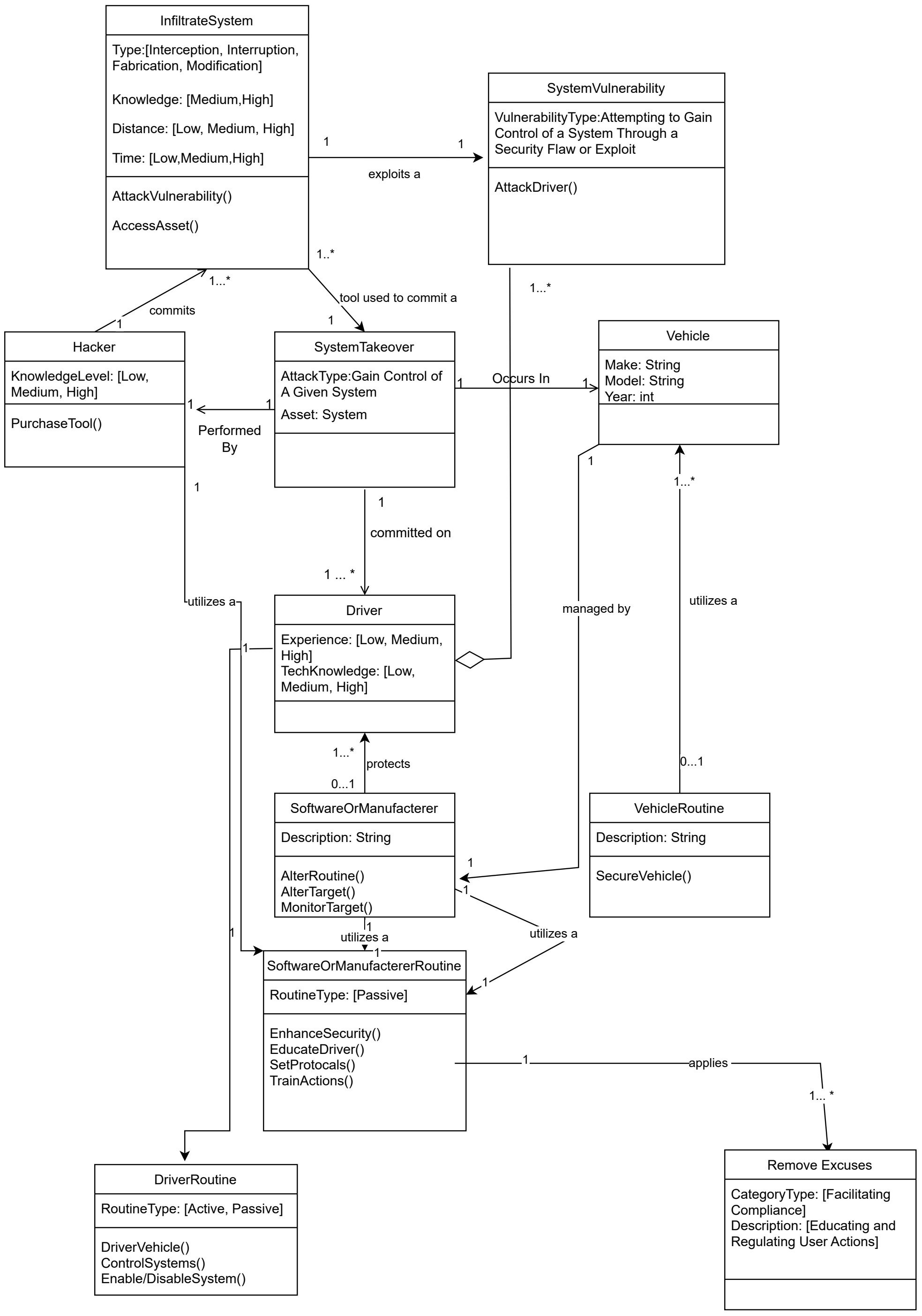


Figure B.3 Gameout

B.3.10 Implementation

This can be done through online education tools, email security checks, or requiring a level of cybersecurity educating when purchasing these autonomous vehicles.

B.3.11 Known Uses

The area of using training to help facilitate emergency situations is common. The use of practiced responses have been done in a traditional sense, when referring to fire drills, tornado drills, and active shooter drills. For example, using online training to help staff in a hospital learn fire drill regulations was used and shown to improve performance [128]. Through online training of a Gameout technique, it could be possible to train stakeholders online for handling a real-world scenario with better results. With reference to cybersecurity, many companies implement phishing training in order to prevent attacks and educate their employees on the proper procedure and identification of fake emails [129]. Furthermore, researchers have found that systems such as PhishGuru have shown to have positive results for assisting users in avoiding phishing attacks [129].

B.3.12 Related Patterns

None

B.4 Public Resource Security

The Public Resource Security pattern seeks to prevent unintended access to user data or system resources via unsecured networks found in a public setting. This can include at the dealership, on-site for a software development company, or around a vehicle with network connectivity.

B.4.1 Pattern Name and Classification

The Public Resource Security pattern is a structural pattern due to the policy changes needed for changing public access rules at a location.

B.4.2 Intent

Public Resource Security provides a pattern for securing public access resources, such as wireless networks at various businesses or locations. This can include password protection on open WiFi, employees using protected-only WiFi, or routine changes to WiFi password. The goal is to prevent easy access to other user's data when using a network. This can help secure information by having a secure network for employees and an open network for visitors or guests. The SCP strategies used by this pattern can be seen in Table B.7

Table B.7 SCP Strategies for Public Resource Security

Increase the Effort	**Increase Risks**	**Reduce Rewards**	**Remove Excuses**
Target Hardening	*Entry/Exit Screenings*	*Target Removal*	*Rule Setting*
Access Control	*Formal Surveillance*	*Identifying Property*	*Stimulating Conscience*
Deflecting Offenders	*Surveillance by Employees*	*Reducing Temptations*	*Controlling Disinhibition*
Controlling Facilitators	*Natural Surveillance*	*Denying Benefits*	Facilitating Compliance

B.4.3 Also Known As

This is known as creating secure networks and secure Wifi. This can also include some firewall information to secure a network.

B.4.4 Motivation

The problem being addressed by the Public Resource Security pattern is to enhance security at a public location to prevent eavesdropping attacks on customers and employees. The solution to this problem will be making sure all networks where sensitive data can be found are properly secured.

B.4.5 Applicability

Public Resource Security is applicable to attack Prevention. Through preventing unwanted manipulation at public resources, we are preventing possible attacks.

B.4.6 Participants

The following field describes the participating classes in the pattern structure.

DataBreach: This represents the crime of DataBreach that attempts to gain control of systems by targeting the asset of information and credentials.

ConnectedUser: This represents the target of a DataBreach that focuses on attacking a specific ConnectedUser. The ConnectedUser may have a level of technical experience.

ConnectedUserRoutine: This represents the routine of a ConnectedUser that focuses on the actions taken. This will have either an active or a passive routine type, and can take actions such as connect devices, login to accounts, manage those accounts, or browse the internet.

Network: This represents the environment where the crime takes place, in this case the Network. The Network may have attributes such as a location, network type, and number of devices connected.

NetworkRoutine: This represents the routine of the Network. The NetworkRoutine can take actions, including connecting users, sending data, and recieving data.

NetworkManager: This entity is the caretaker and guardian for the Network and ConnectedUser. They will have a description, and can take actions to secure the entities it is responsible for.

NetworkManagerRoutine: The set of actions taken by the NetworkManager. This can be either a passive or active routine, and can enhance security, create passwords, set protocols, and monitor the traffic.

ComputerConnectedUser: This represents the offender of a crime as a ComputerConnectedUser with a description of their knowledge level and the ability to monitor a network.

EvasdroppingOnNetwork: This represents the details of an attack that is EvasdroppingOnNetwork used by the ComputerConnectedUser. The type of attack can be interception or modification, will require a low to medium knowledge, and distance will be low as you need to be connected to the network. The time to perform attacks will be low as you can watch these connections in real time along with rapid connection to an open network. The goal with the attack is to focus on a vulnerability and access the asset.

UnsecureNetwork: The UnsecureNetwork that anyone can access to eavesdrop on ConnectedUser to perform a given EvasdroppingOnNetwork.

IncreaseEffort: This is the SCP strategy of IncreaseEffort that will make the ConnectedUser more difficult to attack. This can be done by educating and restricting unauthorized access to make eavesdropping more difficult.

B.4.7 Collaborations

This describes the relationships between entities.

DataBreach: The DataBreach requires a ComputerConnectedUser to attack a ConnectedUser inside the Network. The EvasdroppingOnNetwork is the manner in which the DataBreach is performed.

ConnectedUser: This represents the target of a DataBreach on a ConnectedUser. ConnectedUser has a set of actions and is managed by a NetworkManager

ConnectedUserRoutine: This represents the set of actions of a ConnectedUser.

Network: The environment where the DataBreach takes place, in this case the Network. The Network has a set of actions and is managed by the NetworkManager.

NetworkRoutine: This represents the set of actions of a Network.

NetworkManager: This is the caretaker and guardian for both the Network and the ConnectedUser. The NetworkManager has a set of actions.

NetworkManagerRoutine: This represents the set of actions of a NetworkManager.

ComputerConnectedUser: The entity committing DataBreach using an EvasdroppingOnNetwork attack.

EvasdroppingOnNetwork: The EvasdroppingOnNetwork is used by the ComputerConnectedUser to perform the DataBreach crime. This also exploits a UnsecureNetwork.

UnsecureNetwork: This is the UnsecureNetwork that is attacked by a ComputerConnectedUser and perform a given EvasdroppingOnNetwork.

IncreaseEffort: This SCP strategy should be applied to the NetworkManager.

B.4.8 Structure

The relevant class and sequence UML diagrams can be found below in Figure B.4 for the Public Resource Security pattern.

B.4.9 Consequences

The constraints of this are having multiple people on a public network where other users are unsecured. The secure network would remain safe, however, other users in a building using the unsecured network may not be able to protect valuable information. Table B.8 shows the information for the consequences.

Table B.8 Consequences for Public Resource Security.

Accountability:	The NetworkManager responsible for security would have more accountability.
Confidentiality:	This increases confidentiality of information.
Integrity:	N/A
Availability:	The Network would not be as readily available to Users if it is password protected.
Performance:	The Network requiring both a secure and unsecured network may affect performance.
Cost:	The Network may need additional hardware.
Manageability:	The NetworkManager would need to observe connections more closely.
Usability:	The Network would be less readily available for all Users with password protected aspects.

B.4.10 Implementation

This can be done through altering passwords, creating both a secure and unsecured network, and requiring password changes and the company employees to remain only on the secure network.

B.4.11 Known Uses

The design pattern of Public Resource Security shows the need for secure Wifi as many company's still use unsecured networks [130]. The damage of lack of secure Wifi can cause has

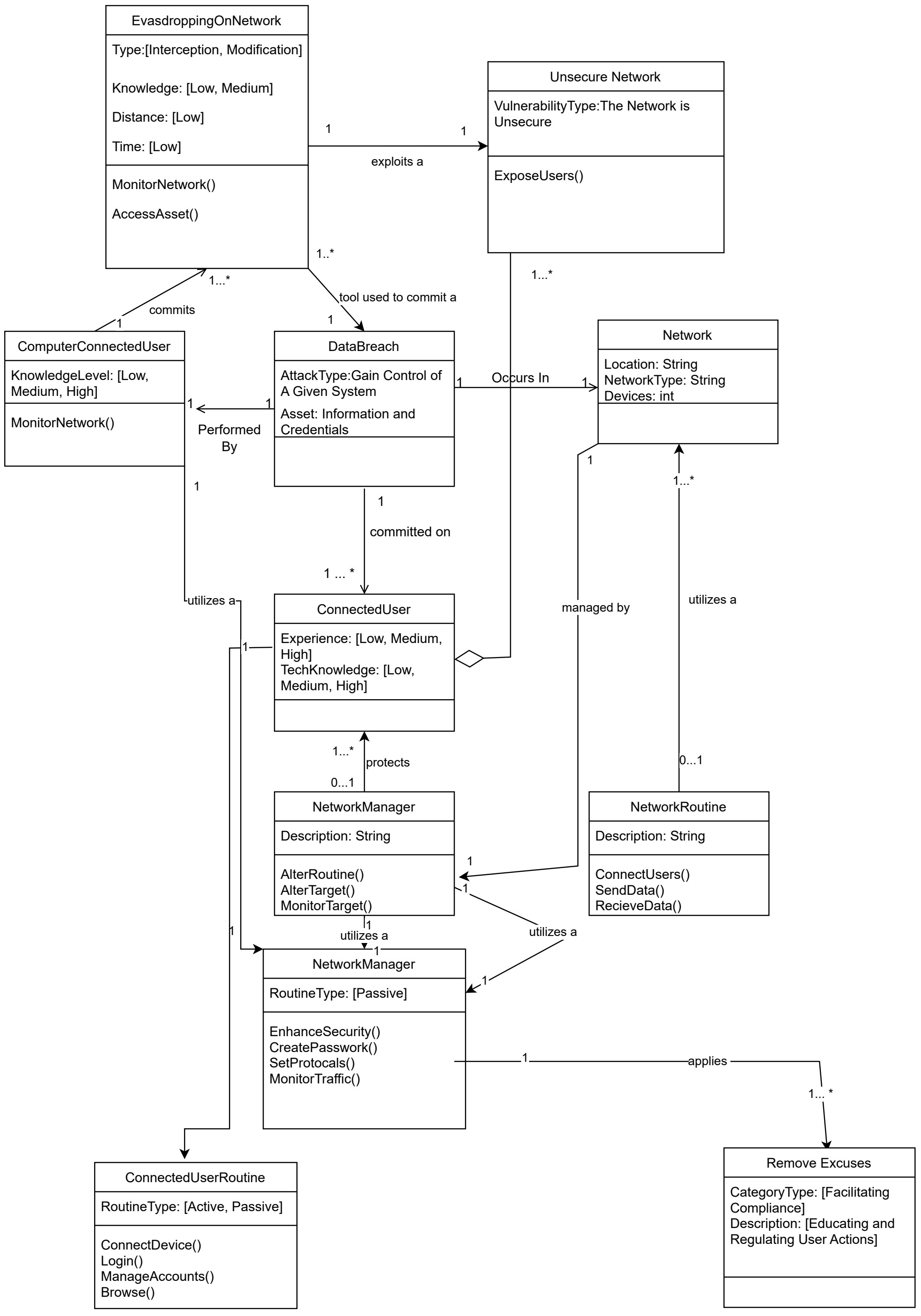


Figure B.4 Public Resource Security

been shown causing vulnerabilities and have practical solutions such as secure systems, updated networks, and updated protocol [131].

B.4.12 Related Patterns

None

B.5 Forum Monitoring

The Forum Monitoring pattern uses moderators via online forums to report, flag, or remove posts that may share information about malicious attacks or vulnerabilities on autonomous vehicles. The early detection of these attacks will enable companies to respond accordingly to growing threats around a vulnerability.

B.5.1 Pattern Name and Classification

This policy is a structural change requiring entities to create a set of rules, regulations, and guidelines to monitor different aspects of communication via the online platform.

B.5.2 Intent

This pattern provides a structure to encourage monitoring and communication with the auto-enthusiasts community using online forums. The existence of motivated enthusiasts outside of the manufacturing companies currently exist. Through websites like Lx Fourms [132] and Jaguar Forums [133], users can post questions and information about information inside a given vehicle. The goal is to monitor these groups communication to assist in finding potential issues, bugs, and other points of concern inside autonomous vehicles. Furthermore, limiting the spread of harmful information for malicious purposes on these platforms. The SCP strategies intended for this pattern can be found in TableB.9.

B.5.3 Also Known As

This pattern also refers to aspects of community involved security.

B.5.4 Motivation

The problem being addressed by the pattern is danger of illicit activity's, discovery of vulnerabilities, and creating a safe online community within autonomous vehicles. This pattern encourages

Table B.9 SCP Strategies for Forum Monitoring

Increase the Effort	**Increase Risks**	**Reduce Rewards**	**Remove Excuses**
Target Hardening	*Entry/Exit Screenings*	*Target Removal*	*Rule Setting*
Access Control	Formal Surveillance	*Identifying Property*	*Stimulating Conscience*
Deflecting Offenders	*Surveillance by Employees*	*Reducing Temptations*	*Controlling Disinhibition*
Controlling Facilitators	*Natural Surveillance*	*Denying Benefits*	*Facilitating Compliance*

a positive and active communication between the company and the community monitoring online forums. The solution strategy is to encourage admins to report behavior that may negatively impact automotive vehicles through encouraging and monitoring an online community dedicated to safety.

B.5.5 Applicability

Forum Monitoring is applicable to prevention, detection, and mitigation. The applicability can vary based on the information shared via online platforms with user engagement, but the main goal is threat prevention and detection.

B.5.6 Participants

The following field describes the participating classes in the pattern structure.

SharingMaliciousHack: This represents the crime of SharingMaliciousHacks that may target various assets.

VehicleSystem: This represents the target of a SharingMaliciousHack that focuses on attacking a specific VehicleSystem. The VehicleSystem can perform a given function.

VehicleSystemRoutine: This represents the routine of a VehicleSystem that focuses on the actions taken. This will have either an active or a passive routine type, and can do things such as perform function, monitor access, and enable/disable the system.

Vehicle: This represents the environment where the crime takes place, in this case the Vehicle. The Vehicle may have attributes such as a make, model, and year.

VehicleRoutine: This represents the routine of the Vehicle. The VehicleRoutine can take actions, including securing the vehicle.

WebsiteOwner: This entity is the caretaker and guardian for both the Vehicle and the VehicleSystem. They will have a description, and can take actions to secure the entities it is responsible for.

WebsiteOwnerRoutine: The set of actions taken by the WebsiteOwner. This can be either a passive or active routine, and will set terms of agreement, create forums, monitor users, and enhance security.

ForumUser: This represents the offender of a crime as a ForumUser with a description of their knowledge level and the ability to communicate with other users.

ForumUserRoutine: The set of actions taken by the ForumUser. This can be either a passive or active routine, and will set login, make posts, respond to posts, and communicate with other users.

SystemExploit: This represents the details of an attack that is a SystemExploit used by the ForumUser. The type of attack can be any and will require a medium to high knowledge. The distance and time will be determined based on the attack. The goal with the attack is to focus on a vulnerability and access a described asset.

UnknownVulernability: This is the UnknownVulernability that attacks a VehicleSystem and perform a given SystemExploit.

IncreaseRisk: This is the SCP strategy of IncreaseRisk that have formal surveillance in an online setting. This can be done by tracking, monitoring, and restricting messages and dangerous inquires on the website.

B.5.7 Collaborations

This describes the relationships between entities.

SharingMaliciousHack: The crime of SharingMaliciousHack requires a ForumUser to attack a VehicleSubsystem inside the Vehicle. The SystemExploit is the manner in which the SharingMaliciousHack is performed.

VehicleSystem: This represents the target of an SharingMaliciousHack on a VehicleSystem. The VehicleSystem has a set of actions and is managed by a WebsiteOwner

VehicleSystemRoutine: This represents the set of actions of a VehicleSystem.

Vehicle: The environment where the SharingMaliciousHack takes place, in this case the Vehicle. The Vehicle has a set of actions and is managed by the WebsiteOwner.

VehicleRoutine: This represents the set of actions of a Vehicle.

WebsiteOwner: This is the caretaker and guardian for both the Vehicle and the VehicleSystem. The WebsiteOwner has a set of actions.

WebsiteOwnerRoutine: This represents the set of actions of a WebsiteOwner.

ForumUser: The entity committing the SharingMaliciousHack using a SystemExploit. The ForumUser has a set of actions.

ForumUserRoutine: This represents the set of actions of a ForumUser.

SystemExploit: The SystemExploit is used by ForumUser to perform the SharingMaliciousHack crime. This also exploits a UnknownVulernability.

UnknownVulernability: This is the UnknownVulernability that attack a VehicleSystem and perform a given SharingMaliciousHack.

IncreaseRisk: This SCP strategy should be applied to the WebsiteOwner.

B.5.8 Structure

The relevant class and sequence UML diagrams can be found in Figure B.5 for the Forum Monitoring pattern.

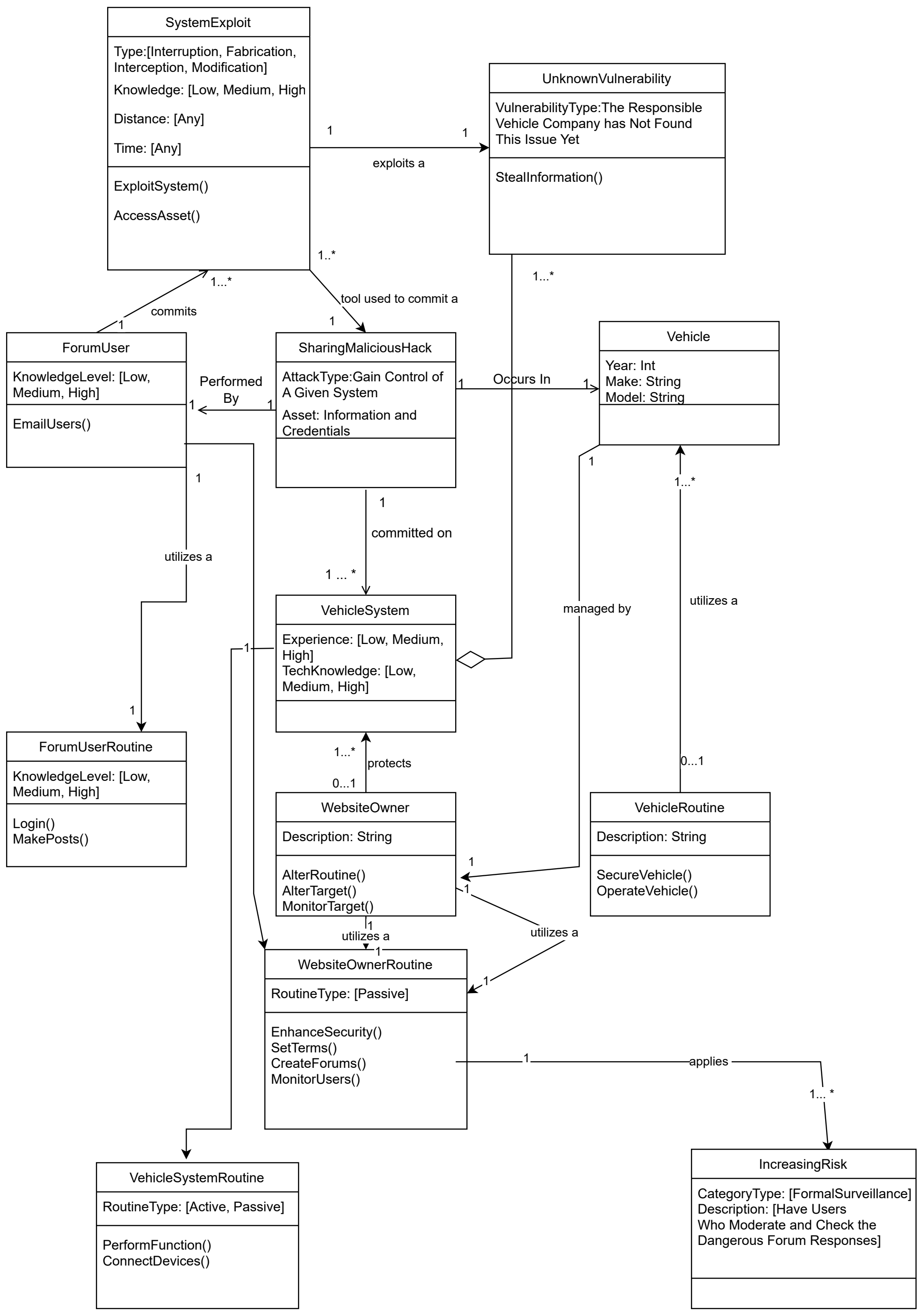


Figure B.5 Forum Monitoring

B.5.9 Consequences

This following describes consequences in implementing the pattern, see Table B.10.

Table B.10 Consequences for Forum Monitoring.

Accountability:	The WebsiteOwner would have moderators who are required to observe online forum communities.
Confidentiality:	The WebsieOwner would have more employees with additional privileges and powers.
Integrity:	N/A
Availability:	The WebsiteOwner may affect User availablity if banned.
Performance:	N/A
Cost:	The WebsiteOwner may need additional positions and employees that could incur costs.
Manageability:	The WebsiteOwner has more employees to observe.
Usability:	N/A

B.5.10 Implementation

This pattern can be executed by partnering with relevant websites or online postings.

B.5.11 Known Uses

In the digital space, the concept of community watch or "neighborhood watch" has been adopted in systems for using open-source software to help secure platforms [134]. The use of online forms in related areas has proven useful for monitoring food safety through data mining [135] and pharmaceutical drug reports via Twitter [136]. In a traditional crime setting, the success of community watch has been shown to be effective for reducing a level of crime [137].

B.5.12 Related Patterns

This has a related pattern of Crowdsourcing.