2019 IEEE Intelligent Vehicles Symposium (IV)
Paris, France. June 9-12, 2019

A Test-Driven Approach for Security Designs of Automated Vehicles

Dajiang Suo, Sanjay E. Sarma

Abstract—**The testing of cyber-physical systems such as automated vehicles (AV) is difficult as engineers face challenges from both cybersecurity and safety domains that start to converge. For cybersecurity, conducting vulnerability testing even before mitigation designs are fixed requires the predication and modeling of adversaries' malicious behaviors. For safety, complete testing at system-level is time-consuming and also infeasible due to the large combination of operational domains. To help engineers design cost-effective mitigation solutions, this paper presents a framework for constructing testing scenarios driven by cyber threats that can be evaluated early in the design process. The testing results can inform the design of mitigation strategies and help engineers in constructing security requirements such that the large solution space will converge more quickly on effective designs. We also illustrate how to build visualization tools to support this process.**

I. Introduction

The testing of cyber-physical systems such as automated vehicles (AV) is difficult as engineers face challenges from both cybersecurity and safety domains that start to converge. For cybersecurity, conducting vulnerability testing even before mitigation designs are fixed requires the predication and selection of adversaries' malicious. Although previously industrial guidebook such as J3061 documents the general procedure of vulnerability testing for automotive systems [1], determining how each cyber threat causes the loss of functionalities (i.e., engineering traceability) and choosing among all threats the most severe ones for early evaluation are still an open-ended question. For safety, Koopman and Wagner [3] suggest complete testing at system-level is time-consuming and also infeasible due to the large space for testing NHTSA Level-4 AVs [2]. The Level-4 AV can drive and respond to emergencies by itself within predefined operational design domains (ODDs) [2]. This indicates that security requirements that set constraints for security designs might be incorrect, incomplete or ambiguous during initial stages of product development.

To help engineers come up with cost-effective mitigation solutions more efficiently, this paper presents a test-driven framework for security designs. In the framework, the construction and selection of testing scenarios is driven by cyber threats that can be evaluated early in the design process. The test results can inform the design of mitigation strategies and help engineers derive security requirements such that the large solution space can converge early on effective designs. To illustrate the proposed approach, a case study is conducted on the emergency-braking feature that is a crucial function for automated driving systems. We also illustrate how to build visualization tools to support this process.

II. Existing approaches

Cybersecurity testing [1] shares common procedures with those of general testing of AVs specified in ISO 26262 [4]. The specified testing framework is based on the "V" software development model. The system (vehicle) is decomposed into sub-components and modules (e.g., hardware for sensing or software algorithms) hierarchically. Sub-components that are tested and passed the test at a certain level will then be integrated and tested against specifications at the level above. One of the challenges in testing of Level-4 automated vehicles is to construct scenarios by iterating all possible combinations of variables in ODDs with reasonable costs. Li et al. [5] propose an approach that generates testing tasks by considering the functionalities that an AV should perform under certain scenarios with temporal and space constraints. The authors suggest the use of simulation testing based on a concept named "parallel" transportation (i.e., a simulation platform that "mirror real testing ground in virtual space"). Masuda [6] suggests that requirements, system specifications and user needs be translated into models that can be implemented in simulation software in order to reduce costs. Thorn et al. [7] propose a framework that provides guidance on building test scenarios with a focus on the safety of automated-driving systems. The authors suggest that a testing scenario consists of and can be built upon four parts — behavioral goals, ODD regarding driving environment, object and events and fail-safe strategies, as shown on the second column of Table I. When it comes to testing scenarios for cybersecurity, its unique characteristic—an adversary who is always actively seeking opportunities to damage the target AV and mobility services it supports — requires additional attention on identifying potential cyber threats beforehand. Engineers need to figure out what can go wrong due to malicious activities during the process of receiving, store and process information based on which safety-critical decisions are made. Specifically, we recommend that engineers consider unique characteristics of security testing that can influence the generation of testing scenarios and the design of mitigation solutions. Table I summarizes the similarity and difference between testing for safety and cybersecurity of AVs.

Dajiang Suo is with Massachusetts Institute of Technology, Cambridge, MA 02139 USA (corresponding author, phone: 857-209-1528; e-mail: djsuo@mit.edu).

Sanjay. E. Sarma, is with Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: sesarma@mit.edu).

978-1-7281-0560-4/19/$31.00 ©2019 IEEE

TABLE I. DIFFERENCE IN BUILDING TESTING SCENARIOS BETWEEN SAFETY AND SECURITY

Testing scenario components	Safety	Cybersecurity
Goal	Behavior or Tactical Maneuver: Immediate control-related tasks with time duration of 1- to 10-s range that ADS is executing as part of the test (e.g., lane following, lane change, turning).	Tactical, Strategic or mission: In addition to behaviors at tactical level, also including those at strategic or mission level (> 10 seconds).
Operational Design Domain	ODD: Operating environment (e.g., roadway type, traffic conditions, or environmental conditions).	ODD&Attacker's model: in addition to OOD related to environment, also account for attacker's power and capabilities.
Objects or events	OEDR: 1.identify expected or unexpected events or objects such as Vehicles, pedestrians or infrastructure, etc. 2.Define response strategies to each above.	Objects, events either contained in malicious messages and signals or exist in environment that are spoofed or tampered: The goal for OEDR in the context of cybersecurity is to detect and respond to spoofed or tampered objects or signals.
Abnormal conditions	The information "failure", whether no data, inadequate or latent data.	In addition to information failures, attacks on any components in the automated-driving system that compromise data integrity.
Mitigation strategies	Fail-Safe (FS) or Fail-Operational (FO): The mapping between effects by failures of information from both physical and logical faults and errors and tactical maneuver behaviors and fail-safe behaviors.	Attack-Secure (AS) or Attack-Operational(AO): The information "failure", whether no data, inadequate or latent data, can be maliciously manipulated such that they can get around the defenses.

III. A TEST-DRIVEN APPROACH FOR SECURITY DESIGN

Given the overlaps and differences in testing scenarios discussed in the previous section, we first set up the scope and goal of our approach. First, we are interested in the intersection between safety and cybersecurity, i.e., vulnerabilities in security designs that may be compromised by adversaries to create safety hazards. For this reason, the testing scenarios derived from the proposed approach are either behavior or tactical and often require immediate control tasks, as show in Table I. Second, the main goal of our approach is to build security into designs early in the AV development process, rather than adding it as an afterthought. Therefore, we are more interested in testing methods that are cost-effective and can be applied early to create security requirements before AV concepts are developed and deployed. This differs from the engineering approach for AV testing in [8] that conducts penetration testing after actual systems being built.

The proposed approach requires collaborations among three teams, as shown in Fig. 1. It starts with threat modeling [9] by the safety and security teams, that is, identify high-level cyber threats that compromise safety or other high-level goals

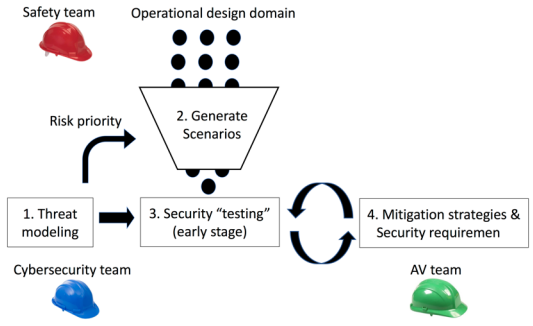


Figure 1. The proposed framework for security testing

for AV designs. This is important for AVs as safety and cybersecurity converge, i.e., cyber attacks on sensing or communication modules can cause incidents and undesired consequences [10]. The results from threat modeling can be stored and sorted based on risk assessment in an attack tree [11]. This is critical for ensuring the traceability between functionalities, cyber threats and mitigation solutions. Based on the risk priority of each threat, engineers from AV design team may select subsets in the target ODDs for constructing testing scenarios. Finally, the three teams jointly decide on the adoption of certain strategies for threat mitigation early in the design process.

A. Threat Modeling

Threat modeling is defined as "a planned activity for identifying and assessing application threats and vulnerabilities in a IT systems" [12]. For cyber-physical systems such as AVs, researchers have been developing models adapted to these systems for identifying cyber threats that cause incidents or undesired losses [1][10][13]. In this paper, we choose the method proposed by Suo et al. [10] for identifying threats on an automated-driving system shown in Fig. 2. In particular, this method of threat modeling provides guidance on identifying cyber threats that leverages Microsoft's STRIDE [14], a model of threats with six categories including Spoofing, Tampering, Repudiation, Information disclosure, Denial of service and Elevation of privilege, and control-theoretic approaches to build the attack tree that maintains the analysis results. The generic model along with detailed processes can be found in [10]. Fig. 3 shows an example attack tree derived from threat modeling analysis.

Our framework for security testing is based on the attack tree derived from Suo's approach [10]. In particular, the traceability among the high-level goals (e.g., safety), system functionalities and cyber threats give clues on which scenarios are critical for testing. Engineers can simulate potential cyber attacks in early stages of product development and evaluate

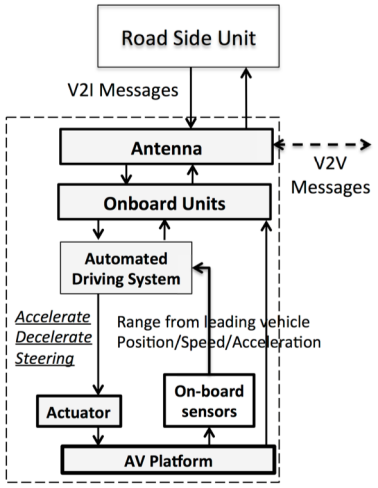


Figure 2. The architecture of an automated-driving system [10]

the effectiveness of certain mitigate strategies without waiting until designs are fixed. For example, the severity of

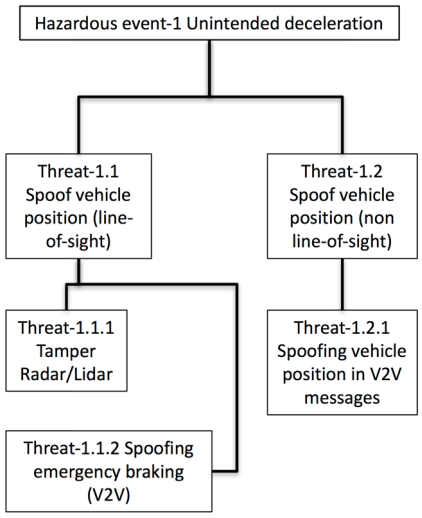


Figure 3. Partial attack tree of automated-driving features [10]

consequences from unintended deceleration command issued by an automated-driving system differs between line-of-sight and non-line-of-sight or low-visibility scenarios. Detecting spoofing position-information that creates ghost vehicles in front when the target AV is moving at a straight road is relatively easier (or takes less efforts) than designing security when vehicle is taking turns on a curved highway where the object in front is not detectable immediately. Similar problems occur under low-visibility conditions where camera may not work and provide correct signal for sensor fusion algorithms.

B. Constructing Scenarios for Testing

ODD is defined as conditions under which automated-driving features are functioning [2] such as geographical and temporal limitations, road types, traffic conditions and external environment. It is critical in constructing testing cases. It to a large extent decides the effectiveness of mitigation solutions to a certain cyber threat. For example, a security design built upon sensor fusion technologies for filtering out ghost vehicles (or objects) in front might not be able to decide whether a vehicle exists in a non-line-of-sight region. As to the robustness of sensors, cameras used for object detection and classifications often do not work well in insufficient light, while Lidar signals can be weakened under certain weather conditions such as rain, fog or snow, etc.

The main challenge in constructing testing scenarios is to select which subset of ODD will be included. As Koopman suggested [3], limiting possible combinations of operational concepts can reduce the complexity in requirements. The proposed attack-tree based approach can help engineers achieve this goal by sorting out subsets of ODDs that are related to cyber threats with higher risks.

C. Mitigation Solutions During Early-Stage Design

The third part constitutes testing cases is a set of mitigation solutions against the cyber attack that engineers want to explore. The generic model proposed for threat modeling can also help in generating these solutions. Related strategies include making infrastructure in the environment hack-proof, designing technologies for robust sensing and communications, conducting sanity-check on information from multiple channels, or modifying algorithms for decision-making for responding to emergencies due to cyber attacks, etc. For example, in order to deal with adversaries who tamper traffic signs or road infrastructure in the environment, engineers can encode digital code into traffic signs [15] for authenticating the identity of these signs, while information from multiple channels (e.g., digital map and camera) can be combined to infer the likelihood that a stop sign appears in a particular location.

A. Tool support for requirement management

Maintaining traceability among engineering activities in Fig. 1 is critical for ensuring the consistency between security requirements and mitigation solutions. We leverage Systems Modeling Language (i.e., SysML) to build a visualization tool to support this task. Fig. 4 shows part of the class diagram in the SysML model we built. It can be divided into three parts —classes for threat modeling, operational design domains for constructing testing cases, and mitigation solutions.

The *Attack Tree* class on the left consists of safety-related events that are either hazardous or undesired, nodes representing cyber threat, and links that connects those events and nodes. Each threat node has a risk rating and belongs to a category in the STRIDE model. As for security testing, the classes of ODD taxonomy and ODD condition on the right in Fig. are critical for building testing cases. For example, the road type (ODD taxonomy) may include one-way highway, two-way highway or city street (ODD condition). The mitigation solutions are associated with one or multiple threats in our SystML model. This is understandable as one cyber threat may be mitigated by more than one solutions, and vice versa. For example, the automated-driving system may detect spoofed vehicle positions included V2V messages based on either sensor fusion techniques or plausibility checks on V2V messages from a single wireless channel.

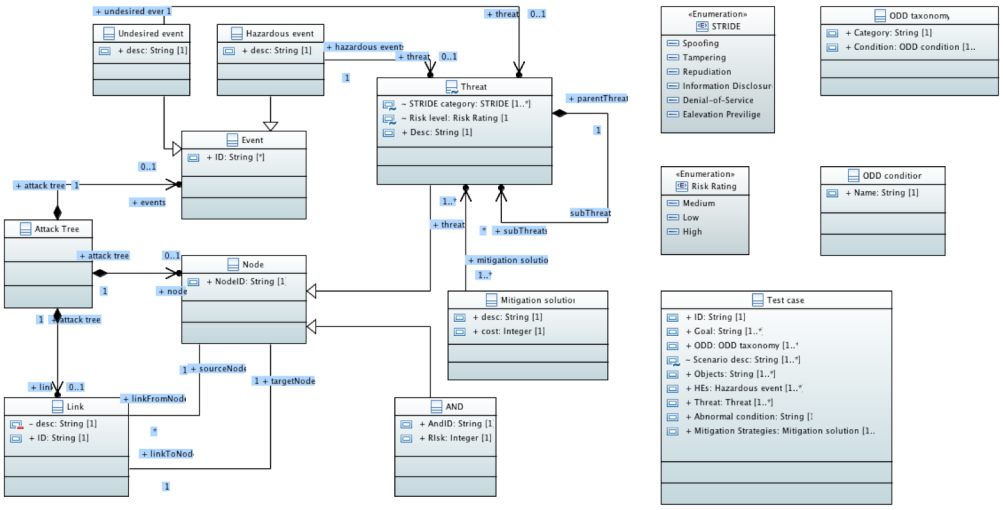


Figure 4. SysML model for supporting the security frameowrk in Fig. 1

IV. A Case Study of Connected And Automated Vehicles

We applied the proposed testing framework to the security design of an AV in this section, as shown in Fig. 2. The architecture of the target AV is created by Suo and Sarma [10] for an illustrative purpose and does not reflect actually designs of any real-world systems. In particular, the automated-driving system (ADS) is installed with automated-driving features such as emergency braking, lane keeping and lane switching by automatically applying acceleration, deceleration and steering, etc. The driving decisions are made based on inputs received from on-board sensors such as camera, radar or Lidar and communication module that support vehicle-to-vehicle (V2V) or vehicle-to-infrastructure (V2I) communications. The inputs include kinematics information of itself including position, speed and acceleration and the relative distance from nearby vehicles, especially the leading ones.

Potential cyber threats on the target AV that cause safety hazards or undesired losses are first given without describing the detailed process, which can be found in [10]. Different types of cyber threats — threats that belong to various categories in STRIDE—can not only cause safety hazards such as unintended deceleration, but also result in other losses that include compromising the serviceability of mobility-sharing supported by AV fleet or the emergency-response capabilities of transportation authorities. We focus our discussion on cyber attacks of spoofing vehicle position or identities on on-broad sensors (e.g., radar, Lidar) that lead to unintended deceleration.

A. Scenarios and test cases

After determining the risk-priority of cyber threats in the attack tree, engineers can select those with highest risk priority to build test cases. Table II gives an example of testing scenarios related to the threat of creating ghost (vehicle) identities in front of the target vehicle by spoofing radar signals. The high-level goals include applying controls for vehicle following while maintaining a safe distance at the same time,

as shown in the second row. For selecting ODDs for constructing scenarios, the context information included in safety hazards or undesired events can help them choose subsets of ODD such that the complexity due to combinations of possible operational concepts. For example, for cyber threats resulting in unintended deceleration, a reasonable choice of ODD variables includes speed and road type because they are closely related to the braking function. In addition, engineers may also consider road geometry and illumination or even weather-induced conditions (e.g., fog) as they can affect on-board sensors (e.g., Radar, Lidar, camera, etc.) that provide feedback for emergency-braking decisions. In addition, any nearby object should be included if it can influence the decision-making by automated-driving systems in the selected scenarios (5th row of Table II).

B. Designing mitigation solutions

Each test case constructed before will be evaluated under attack scenarios. However, the purpose of testing here is to address cyber threats early in the design process by iterating mitigation strategies [23]. In this regard, any experimental or simulation tools based on which engineers can evaluate the effectiveness of preliminary security designs can serve this purpose. We use *Driving Scenario Designer* included in MATLAB R2018b [19] for illustrating how can design the placement of Lidar or camera for robust sensor fusion under spoofing attacks on on-board sensors. However, engineers may select experiment tools that are most appropriate for the evaluation task. For example, Lu et al. [24] use PreScan simulation platform [25] for evaluating security designs against spoofing attacks on V2V messages for cooperative adaptive cruise control (CACC).

Assuming engineers come up with four strategies as listed below.

- Anti-jamming design of Radar and/or Lidar signal [17][18]

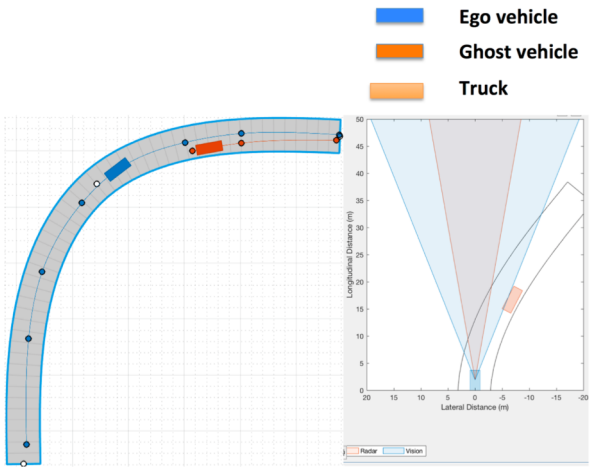
TABLE II. TESTING CASES OF SPOOFING VEHICLE POSITION

Testing cases for cyber threat—Sudden appearance of ghost vehicles (within line-of-sight)	
Goals	•Tactical: vehicle-following with a safe distance •Control: applying longitudinal and lateral control for preventing collisions without compromise vehicle stability and passengers' experience
ODD	Road Type={1-way highway,2-way highway }, Time={daytime/night}, Speed ={45~65mph}, Roadway Geometry={straight, Curved}
Scenarios	•Vehicle moving on a straight highway between the speed of 45 and 60 mph •Vehicle approaching a roundabout •Vehicle moving at a single-lane curved road •Vehicle moving at a two-lane curved road with head-on cars
Object	The front-vehicle in current lane, vehicles in adjacent lanes, traffic signs, lights; construction signs, pedestrians
Abnormal conditions due to cyber attacks	Vehicle sudden appearance indicated by Radar, Lidar signals, or V2V messages
Mitigation strategies	•Plausibility checking on radar/Lidar data (preprocessing of receiver data) •(Cross-channel) sanity checking by sensor fusion •Plausibility checking on post-processing data by sensor fusion •Anti-spoofing design of Radar signal

- Plausibility checking on radar/lidar data (preprocessing of receiver data)
- (Cross-channel) sanity checking by sensor fusion
- Plausibility checking on post-processing data by sensor fusion

Each strategy will be tested against a set of scenarios included in the test case. Fig. 5 gives two example scenarios synthesized by using the *Driving Scenario Designer* included in MATLAB R2018b [19]. Fig. 5a shows a highway road in which the ego car (blue color) moving at the speed between 40 and 65 miles per hour detect a (ghost) vehicle (orange color) right after getting out of the corner. A more complicated scenario is described in Fig. 5b in which the ego vehicle for testing is moving at a two-lane highway in which an adversary is trying to spoof a head-on car that is overtaking a truck. It should be addressed though that this paper does not intend to explore the pros or cons of different simulation software.

This example illustrates that minor variations in a given scenario can make the original security solutions invalid. Consider engineers make tradeoffs between the second or the third mitigation strategy. The second strategy adds plausibility checking logic into decision-making algorithms of the ego car. This strategy may work for straight-lane scenarios but not necessarily work for scenarios involving curve lanes. For example, comparing the current position of the leading vehicle indicated by the ego car's own radar data with the predicated position based on the leading vehicle's trajectory [20] is based on the assumption that the relative rate of approaching the leading vehicle is less than a certain threshold such that the ego vehicle's data pre-process module is able to collect enough detections until it confirms that the signal represents a real car rather than a ghost. However, the



a. Curved road with single lane

b. Curved road with multiple lanes and head on vehicles

Figure 5. Examples of synthesized scenarios in which a (ghost) vehicle suddenly appearances in front of the ego-vehicle

assumption may not hold in the case that a head on vehicle approaches in a curve. In this case, the radar may only detect a vehicle that suddenly appears without collecting enough evidence. If this happens, the tracking logic will not be able to differentiate the head on vehicle from a ghost vehicle spoofed by an adversary. Methods used for filtering out radar clutter for ensuring safety [22] such as increasing the confirmation threshold of vehicle detection in tracking history [21] will not work for security-related scenario.

The third strategy adopts sensor fusion of radar (Lidar) and camera for sanity check on data from multiple channels (third mitigation strategy) may require special designs on the placement of radar. Fig. 6 (top-right) shows the placement of two radars on each side of the ego vehicle to increase the field of view for the radar system such that it

can detect the leading vehicle or even head-on vehicles earlier than the original design. The detection information can be combined with camera's inputs. In this way, the tracking algorithm can collect enough information about the trajectory history to confirm whether the indicated leading vehicle is caused by malicious inputs from Radar or not.

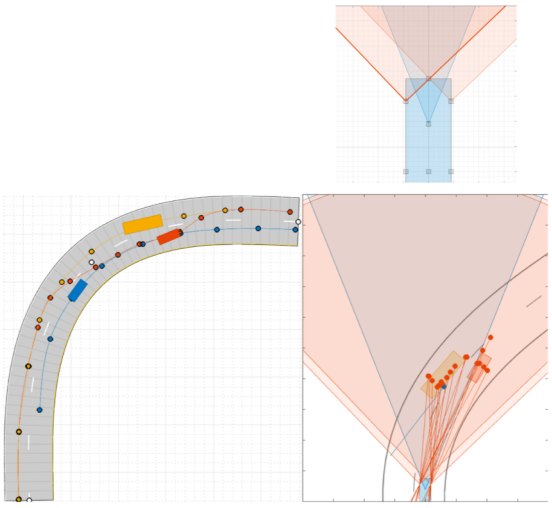


Figure 6. Redesign mitigation solutions by camera placement for sanity checking based on sensor fusion

C. *Discussion*

The complexity of tasks for designing mitigation solutions lies in the fact that engineers need to make trade-offs between ensuring a complete threat coverage, i.e., all (or high risk) cyber threats are mitigated by current security designs, and maintaining a reasonable cost to build and deploy solutions. We build a proof-of-concept tool to visualize the design process based on the SysML model discussed before, as shown in Fig. 7. The automated-driving system (Fig. 2) and its interaction with nearby vehicles is included on the left, while the attack tree corresponding to the hazardous event of unintended deceleration (Fig. 3) is shown on the right. One example of designing solutions to mitigate spoofing vehicle positions in V2V messages is given on the bottom of Fig. 7. In addition to conducting sanity checks on V2V messages directly [20], engineers may come up with sensor fusion algorithms to filter out malicious V2V messages if the signal of vehicle positions received from on-board sensors such as Lidar and camera has higher confidence levels than wireless communication [16].

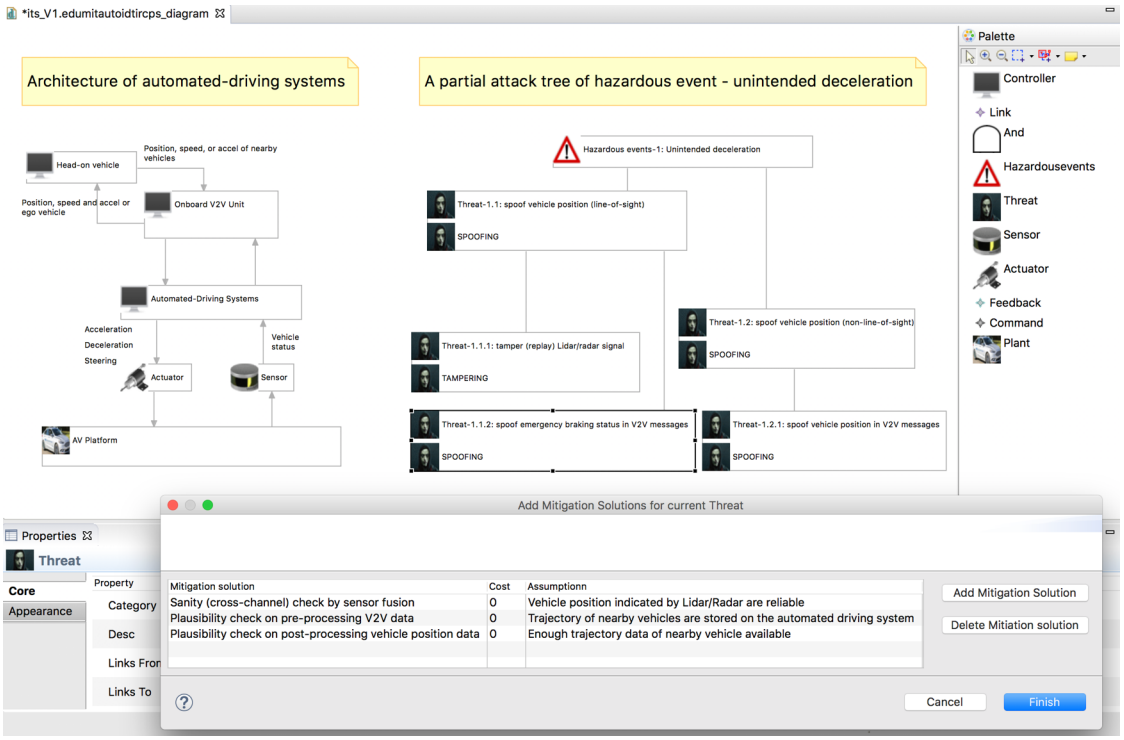


Figure 7. User interfaces of the visualization tool for supporting security testing. This tool is developed from the SysML model introduced in this paper

V. Conclusion

A framework for security testing of NHTSA Level-4 AVs is presented to support the design of mitigation strategies early in the design process. The proposed approach builds upon a threat modeling technique developed for cyber-physical systems and provides guidance on choosing ODDs for constructing testing scenarios.

While enabling early evaluation of mitigation strategies, the proposed approach has limitations that require more research efforts in the future. First, evaluating the effectiveness of mitigation strategies in the vehicle-level (e.g., sensor fusion algorithms) requires an integrated environment that can simulate cyber threats from both on-board sensors and V2X modules (e.g., DSRC). To the best of the author's knowledge, such integrated simulation platforms are still lacking. Second, many cyber threats on cyber-physical systems such as spoofing signals in the control area network (CAN) or sensors for monitoring vehicle status like tire-pressure monitoring modules are hardware-related and cannot be simulated in software environment.

References

[1] SAE International: J3061 Cybersecurity Guidebook for Cyber-Physical Vehicle Systems, Jan. 2016.

[2] SAE International: J3016 Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles, Sep. 2016.

[3] Koopman, P., & Wagner, M., "Challenges in autonomous vehicle testing and validation," SAE International Journal of Transportation Safety, 4(1), 15-24.

[4] Road vehicles -- Functional Safety -- Part 2: Management of functional safety, ISO 26262-2:2018, Dec. 2018.

[5] Li, L., Huang, W. L., Liu, Y., Zheng, N. N., & Wang, F. Y., "Intelligence testing for autonomous vehicles: a new approach," IEEE Transactions on Intelligent Vehicles, 2016, (2), 158-166..

[6] Masuda, S., "Software Testing Design Techniques Used in Automated Vehicle Simulations," In Software Testing, Verification and Validation Workshops (ICSTW), 2017 IEEE International Conference on (pp. 300-303). IEEE.

[7] Thorn, E., Kimmel, S., Chaka, M., (2018, September). A Framework for Automated Driving System Testable Cases and Scenarios (Report No. DOT HS 812 623). Washington, DC: National Highway Traffic Safety Administration.

[8] Thompson, H. H. (2005). Application penetration testing. IEEE Security & Privacy, 3(1), 66-69.

[9] Swiderski, F., and Snyder W., Threat Modeling, Microsoft Press, 2004.

[10] Suo, D., Siegel, J. E., and Sarma, S. E., "Merging safety and cybersecurity analysis in product design," IET Intelligent Transport Systems, 2018, 12(9), 1103-1109.

[11] Schneier, B. (1999). Attack trees. Dr. Dobb's journal, 24(12), 21-29..

[12] OWASP, https://www.owasp.org/index.php/Category:Threat_Modeling.

[13] Young, W., Leveson, N.G.: 'An integrated approach to safety and security based on systems theory', Commun. ACM, 2014, 57, (2), pp. 31–35.

[14] 'The STRIDE Threat Model', available at https://msdn.microsoft.com/en-us/library/ee823878(v=cs.20).aspx, accessed 15 Jan. 2019.

[15] The quest to design a smart road, https://www.3m.com/3M/en_US/road-safety-us/news-3m-connected-roads/full-story/~the-quest-to-design-a-smarter-road/?storyid=bf172e7a-2f42-43da-b813-977db4e7798f, accessed 05 Feb. 2019.

[16] Bismeyer, N., Mauthofer, S., Bayarou, K.M. and Kargl, F., 2012, November. Assessment of node trustworthiness in vanets using data plausibility checks with particle filters. In Vehicular Networking Conference (VNC), 2012 IEEE (pp. 78-85). IEEE.

[17] Yan, C., Xu, W., & Liu, J., "Can you trust autonomous vehicles: Contactless attacks against sensors of self-driving vehicle," 2016, DEF CON, 24.

[18] Petit, J., Stottelaar, B., Feiri, M., & Kargl, F., "Remote attacks on automated vehicles sensors: Experiments on camera and lidar," Black Hat Europe, 11, 2015.

[19] MathWorks: Driving Scenario Designer, available at: https://www.mathworks.com/help/driving/ref/drivingscenariodesigner-app.html

[20] Schmidt, R.K., Leinmüller, T., Schoch, E., Held, A. and Schäfer, G., 2008, June. Vehicle behavior analysis to enhance security in vanets. In Proceedings of the 4th IEEE Vehicle-to-Vehicle Communications Workshop (V2VCOM2008).

[21] MathWorks: Introduction to Track Logic, available at: https://www.mathworks.com/help/fusion/examples/introduction-to-history-based-and-score-based-track-logic.html.

[22] Park SW., "Design and test of traffic jam assist, A Case Study," available at: https://www.mathworks.com/content/dam/mathworks/mathworks-dot-com/company/events/conferences/automotive-conference-michigan/2018/proceedings/design-and-test-traffic-jam-assist.pdf.

[23] W. Young, and N. G. Leveson. "An integrated approach to safety and security based on systems theory." Commun. ACM 57, no. 2 (2014): 31-35.

[24] P. Lu, L. Zhang, B. Park, and L. Feng, "Attack-Resilient Sensor Fusion for Cooperative Adaptive Cruise Control," In 2018 21st International Conference on Intelligent Transportation Systems (ITSC), pp. 3955-3960. IEEE, 2018.

[25] "Prescan simulation platform," https://tass.plm.automation.siemens.com/prescan.