A System Theoretic Approach to Cybersecurity Risk Analysis and Mitigation for Autonomous Passenger Vehicles

Chee Wei Lee and Stuart Madnick

Working Paper CISL# 2018-09

February 2018

Cybersecurity Interdisciplinary Systems Laboratory (CISL)
Sloan School of Management, Room E62-422
Massachusetts Institute of Technology
Cambridge, MA 02142

A System Theoretic Approach to Cybersecurity Risk Analysis and Mitigation for Autonomous Passenger Vehicles

Chee Wei Lee

Stuart Madnick

February 2018

MIT Sloan School of Management

Abstract

Urban Mobility is in the midst of a revolution, driven by the convergence of technologies such as artificial intelligence, on-demand ride services, as well as connected and self-driving vehicles. Technological advancements often lead to new hazards and changing nature in how accidents can happen. Coupled with the increased levels of automation and connectivity in the new generation of autonomous vehicles, cybersecurity is emerging as one of the key threats affecting the safety of these vehicles. Traditional hazards analysis methods treat safety and security in isolation, and are limited in their ability to account for interactions among organizational, socio-technical, human, and technical components. In response to these challenges, the System Theoretic Process Analysis (STPA) was developed to meet the growing need for system engineers to holistically analyze complex socio-technical systems.

We applied STPA-Sec, an extension to STPA to include security analysis, to co-analyze safety and security hazards, as well as identify mitigation requirements. The results were compared with another promising method known as Combined Harm Analysis of Safety and Security for Information Systems (CHASSIS). Both methods were applied to the Mobility-as-a-Service use case, focusing on over-the-air software updates feature. Overall, STPA-Sec identified additional hazards and more effective requirements compared to CHASSIS. In particular, STPA-Sec demonstrated the ability to identify hazards due to unsafe/ unsecure interactions among sociotechnical components. This research also suggested using CHASSIS methods for information lifecycle analysis to complement and generate additional considerations for STPA-Sec. Finally, results from both methods were back-tested against a past cyber hack on a vehicular system, and we found that recommendations from STPA-Sec were likely to mitigate the risks of the incident.

Keywords: Cybersecurity, Autonomous Vehicles, Risk Analysis, Mobility-as-a-Service, STPA-Sec, System Theoretic Process Analysis, Cybersecurity Hazards Analysis

Table of Contents

Autonomous vehicles for urban mobility

State of the art

Mobility-as-a-Service (MAAS) is a fleet of autonomous, self-driving vehicles for ride-sharing services. A concept widely perceived as the future of urban transportation, MAAS is expected to radically change the car ownership model. Based on ARK's research [1], the global MAAS revenue will exceed $10 trillion in gross revenue by 2030, roughly ten times the market for autonomous vehicles sale (see **Figure 1**). Companies such as Uber, Tesla, and nuTonomy have on-going efforts to develop autonomous vehicles as ride-sharing service similar to the MAAS. **Figure 2** shows the generic architecture for the MAAS, which comprises the autonomous vehicles, backend cloud infrastructure, as well as devices connected to the cloud.

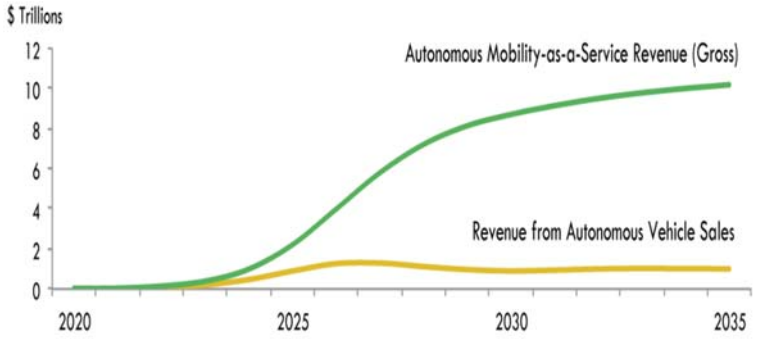

Figure 1. Global revenue for autonomous cars and services [1]

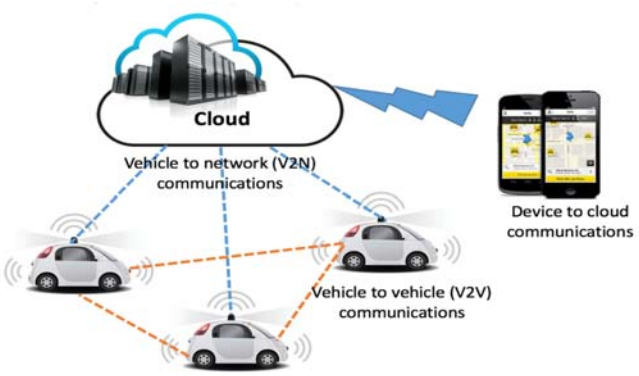

Figure 2. Generic architecture for connected autonomous vehicles in MAAS

Evolving cybersecurity threats and impacts

As cyber-physical systems (CPS) in autonomous vehicles get more sophisticated, new threats are beginning to surface, making safety and security analysis more challenging, as exemplified by the following incidents involving automotive:

- Oct 18, 2016, Singapore: A driverless vehicle developed by nuTonomy was involved in a minor accident with a lorry (see **Figure 3**). nuTonomy's internal investigations concluded that the incident was due to "an extremely rare combination of software anomalies" [2]. Although this was a minor accident with no personnel injuries and did not involve a cyber attack, it shows how accidents can occur due to unexpected interactions between software components that may individually be working perfectly well.

Figure 3. Minor accident involving nuTonomy AV in Singapore

- May 7, 2016, Florida: The first known fatal accident involving a semi-autonomous vehicle, Tesla S70 collided with the side of a tractor-trailer (see **Figure 4**), resulting in the death of the driver behind the wheels of the S70. According to Tesla, "the high, white side of the box truck" — that apparently caused the system to believe the truck was an overhead sign — "combined with a radar signature that would have looked very similar to an overhead sign, caused automatic braking not to fire" [3].

Figure 4. Fatal accident involving Tesla semi-autonomous vehicle in Florida

- 2015 – 2016: Two cybersecurity researchers demonstrated that they were able to remotely control key features of the Jeep Cherokee, including its steering, braking, transmission, and brakes (see **Figure 5**). The researchers exploited the vehicle's infotainment features to remotely plant vulnerabilities into the electronic control unit (ECU). Although we have not seen such cyberattacks leading to accidents on the roads, there has been numerous experiments to demonstrate the vulnerabilities of communication device in connected vehicles [4] [5].

Figure 5. Remote car-jacking of Jeep Cherokee

Three key observations can be made from the above incidents. First, advanced features in new generation of autonomous vehicles mean vehicles are increasingly complex and connected, increasing the attack surface for cyber-attacks. Examples of such attacks include installing malicious codes, and remotely taking control of vehicular safety-critical functions. Such attacks can be conducted in large scale with relatively little effort, potentially affecting the safety of passengers and other road users. These could lead to ominous possibilities involving harm to both drivers and pedestrians.

Second, the development and operation of autonomous vehicles have increased coupling among socio-technical components. Safety and security analysis are no longer limited to standalone systems; interactions among components in the larger ecosystem comprising technical, environmental, organizational, managerial, and regulatory aspects must be holistically considered. The accident between the Tesla S70 and tractor-trailer demonstrated how interactions between the vehicle, objects on the road (other vehicles and road signs), and the environment (bright, sunny conditions) can lead to an accident.

Third, the pace of technology advancement and pressure to reduce the time to market means developers have limited time to fully understand potential behaviours and risks before systems become operationalized. Furthermore, the nature in which hazards and accidents occur continue to evolve, leading to limited ability to learn from past knowledge and experiences. Also, especially in the case of cyber attacks, the skills and approaches of attackers continue to change.

Traditional analysis methods that aim to assess the safety of critical infrastructures are limited in their ability to encompass the complexity of such emerging CPS. Independent studies have also shown strong mutual influence between safety and security aspects [6]. To address the above challenges, a holistic approach to co-analyze safety and security risks is necessary with the emergence of the next generation of passenger autonomous vehicle. Systems-Theoretic Process Analysis (STPA), a deductive hazards analysis methodology based on systems theory, was developed by Nancy Leveson at MIT [7] to address this type of need. Compared with traditional methods designed to prevent component failures, STPA also addresses component interaction accidents that can arise from design flaws, dysfunctional interactions or unsafe control actions. This research applies STPA-Sec, an extension from STPA from safety analysis to cybersecurity analysis, to identify potential risk areas and mitigations for passenger autonomous vehicle. The findings from STPA-Sec were compared with another hazards analysis method – Combined Harm Analysis for Safety and Security for Information Systems (CHASSIS) – to identify strengths and weaknesses in both approaches.

Safety and cybersecurity analysis methods

Traditional methods for safety hazards analysis include Failure Mode and Effect Analysis (FMEA) detailed in [8] and Fault Tree Analysis (FTA) detailed in [9]. Both FMEA and FTA have been widely used in various industries to analyze safety hazards and derive safety functional requirements. However, they do not specifically cover cybersecurity hazards analysis.

Recognizing the tight interplay between safety and security, combining safety and security hazards analysis in the engineering process has become a new interesting research topic in recent years [10][11]. Multiple approaches have been developed to support co-analysis of safety and security for automotive hazards analysis: (1) SAHARA (A Security-Aware Hazard and Risk Analysis Method) [12] extends the classical hazards and risks analysis with security related guide words and an evaluation of risks; (2) FMVEA (Failure Mode, Vulnerabilities and Effects Analysis) [13] extends FMEA with threat modes and vulnerabilities; (3) CHASSIS (Combined Harm Assessment of Safety and Security for Information Systems) [14] is a methodology for safety and security assessments and formulation of mitigation measures, based on use case and sequence diagram modelling.

In the area of safety and security analysis for automotive, [15] proposes a risk assessment framework or autonomous and cooperative automated driving. The proposed framework adopts the convention of the NHTSA threat model and categorized attack methods using the STRIDE classification: **S**poofing Identity, **T**ampering with Data, **R**epudiation, **I**nformation Disclosure, **D**enial of Service, and **E**levation of Privilege. Each threat is consolidated in a threat matrix (see **Figure 6**) considering the following factors:

- Attack potential (vertical axis): Considers the difference between the threat agent's ability to execute a successful attack and the system's ability to withstand such attacks. Parameters include the time required for an attacker to identify a vulnerability

and launch an attack; availability of attacker's finances versus finances required to launch a successful attack; attacker's skills set versus the system's required skills.

- Motivation (horizontal axis): Considers the motivation and determination of the threat agent to execute the attack. Parameters include financial gain, ideology, passion, and risk.
- Impact (size of circle). Considers the losses to stakeholders in the event of successful attack, factoring financial loss, privacy and safety consequences.

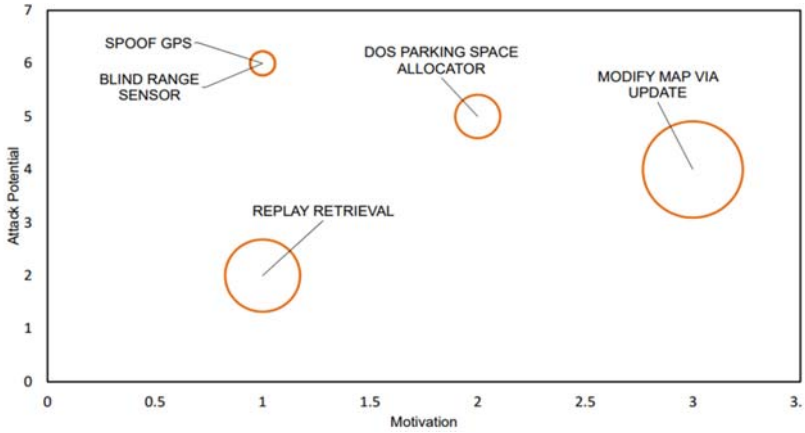

Figure 6. Example threat matrix visualization for driverless valet parking example [15]

System theoretic approach to cybersecurity risk analysis

STAMP theory

Systems-Theoretic Accident Model and Processes (STAMP) was developed by Leveson from MIT after many years of research on safety. The STAMP model of accident causation is built on three basic concepts – safety constraints, a hierarchical safety control structure, and process models – along with basic systems theory concepts [7]. In contrast to tradition methods such of FMEA and FTA that are based on the reliability of individual components, STAMP focuses on the emergent properties of engineered systems and treats safety as a control system problem. STAMP uses system theory to represent the system as hierarchical control structures, where each level imposes constraints on the activity of the level beneath it [7]. This hierarchical structuring allows the system model to capture not only accidents due to component failures and component interactions, but also extends to understanding incomplete or missing requirements from external socio-technical components.

Figure 7 shows a generic hierarchical control structure that includes system development on the left, and system operations on the right. Commands or control actions are given by higher levels of control processes to lower levels throughout the hierarchy, and feedback is provided from lower levels to higher level. Traditional safety hazards analysis typically focuses on the operating process of system components, as shown in the bottom right of the figure. STAMP-based analysis considers control structures that include regulatory, organizational, engineering, and human components, and can therefore analyze additional causal scenarios not included in traditional approaches.

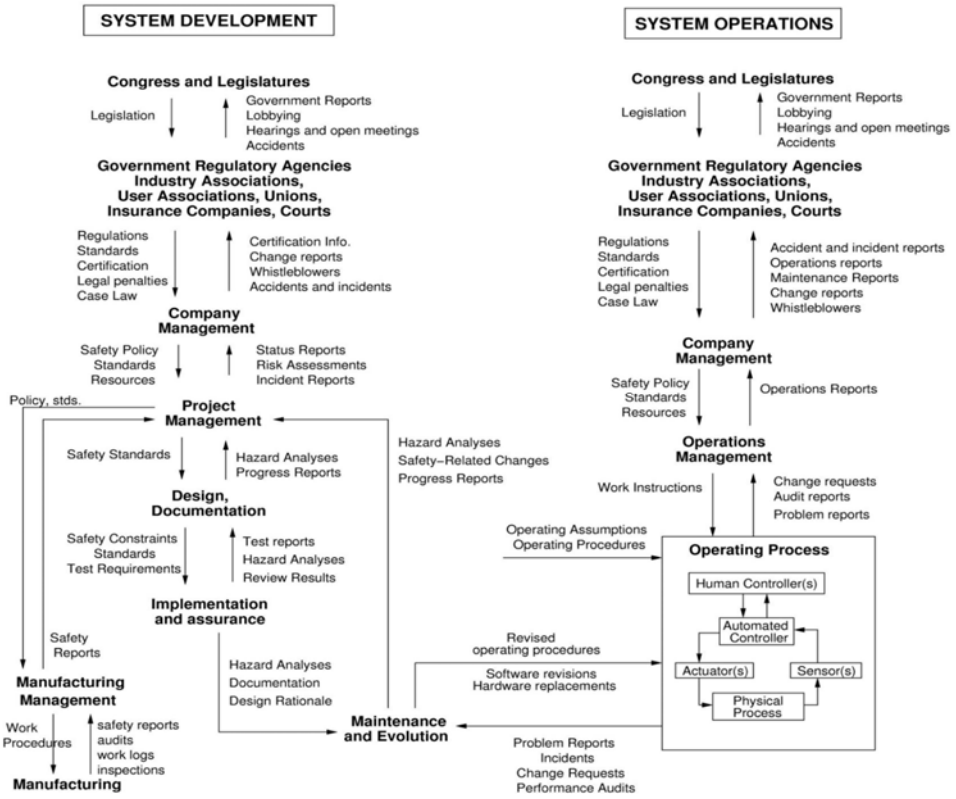

Figure 7: Leveson's general control structure of a social-technical control structure [7]

STPA-Sec: An extension to safety hazards analysis applied to security

STPA is a deductive hazard analysis method based on STAMP and is used to derive requirements for accidents and loss prevention. One of the strengths of STPA is its applicability to early stages of concept development phase.

STPA-Sec extends STPA from safety to cybersecurity analysis, and is used to identify system vulnerabilities and requirements for cyber and cyber-physical systems. Since its inception, STPA has been applied to a wide range of domains ranging from automotive systems, e.g. [16]; automation and workplace safety, e.g. [17]; aviation systems, e.g. [18]; medical devices [19]; and other emergent system properties such as security, e.g. [20]. In [20], Young and Leveson introduced STPA-Sec, suggesting the use of a causality model based on system theory to provide an integrated and more powerful approach to safety and security co-analysis. In recent years, the STAMP-based approach has been applied to manage cybersecurity risks of in various systems: [21] applies STAMP to analyze cyber-attacks on TJX, and revealed insights which had been overlooked in prior investigations; [22] [23] utilizes STAMP-based approach to analyze cyber threats in applied to the Stuxnet case, an attack designed to disrupt the Iranian Nuclear program.

Applying STPA-Sec on Mobility-as-a-Service vehicle fleets

STPA-Sec analysis comprises three key stages as summarized below. While the analysis is presented straight through, there has been a number of iterations in each step as new findings were incorporated to refine the analysis. The three key stages are namely:

- **Establish the system engineering foundation**, which includes defining and framing the problem, as well as identifying accidents / losses and hazards related to the case.

- **Identify potentially unsafe/ unsecure control actions**, which documents the generic functional control structure and control actions that may lead to the identified hazards.
- **Identify causes of unsafe/ unsecure control actions and eliminate or control them**, which includes identification of scenarios leading to unsafe/ unsecure control actions, and using the identified unsafe control actions to create safety requirements and constraints

Stage 1: Establish the system engineering foundation

Stage 1 covers the preliminary steps of STPA-Sec analysis: first to identify the goal/ purpose of the system and then to identify the accidents / losses and hazards related to the system. The key outcome from STPA-Sec is to derive a set of safety and security requirements to eliminate or control unsafe interactions within the control structure. To achieve this, STPA-Sec co-analyses security and safety hazards using a top-down approach, starting from the identification of unacceptable accidents and losses, as well as potential hazards related to the system. The method is further illustrated in the following sections.

Identifying accidents / losses related to the system

Accidents/ losses are defined by Leveson as "An undesirable or unplanned event that results in a loss, including loss of human life or human injury, property damage, environmental pollution, mission loss, etc." [7]. The unacceptable losses and accidents considered in the analysis are[1]:

- A1: Damage to vehicle or public property
- A2: Injury or death to people
- A3: Degradation of system availability or performance
- A4: Loss of critical information

Identifying hazards related to the system

Leveson defines hazards as "A system state or set of conditions that, together with a particular set of worst-case environmental conditions, will lead to an accident (loss)" [7]. Although a system that is in hazardous state does not guarantee that it will lead to an accident, it is crucial to prevent the occurrence of hazards by mitigation through system design or organizational polices and guidelines.

The high-level hazards and associated accidents/ losses identified in our analysis are shown in **Table 1**.

Table 1. Hazards and associated accidents/ losses

Hazards	Associated Accidents/ Losses
H1: Adversaries take over control of safety-critical functions of AV	A1, A2, A3
H2: AV operating with unsafe/ unsecure/ outdated software	A1, A2, A3
H3: Adversaries compromise network/ critical infrastructure supporting AV	A3, A4
H4: AV travelling on unsafe/ unauthorized road	A1, A2

Stage 2: Identify potentially unsafe/ unsecure control actions

Functional Control Structure

Our analysis focuses on the over-the-air (OTA) software updates, a key capability which enables the connected autonomous vehicle to exchange live updates such as traffic/ road conditions, routing instructions and location updates, as well as periodic firmware updates and bug fixes. The ability to receive OTA software updates is key to realising the MAAS concept, but it also poses a different range of attack surfaces that can be exploited.

[1] The numbers, A1, etc., are just used for reference. There is no indication of priority implied.

Figure 8 shows the high-level functional control structure of socio-technical components in the OTA software updates example. The functional control structure details the control loops within the system, together with interactions among components at different hierarchical levels. This functional control structure provides the basis to further analyze safety and security constrains within the system. The system boundary under analysis in this study is represented by components in the shaded box. Although the analysis is limited to components within the system boundary, it is important to consider interactions with external socio-technical systems to glean additional insights and context to the analysis.

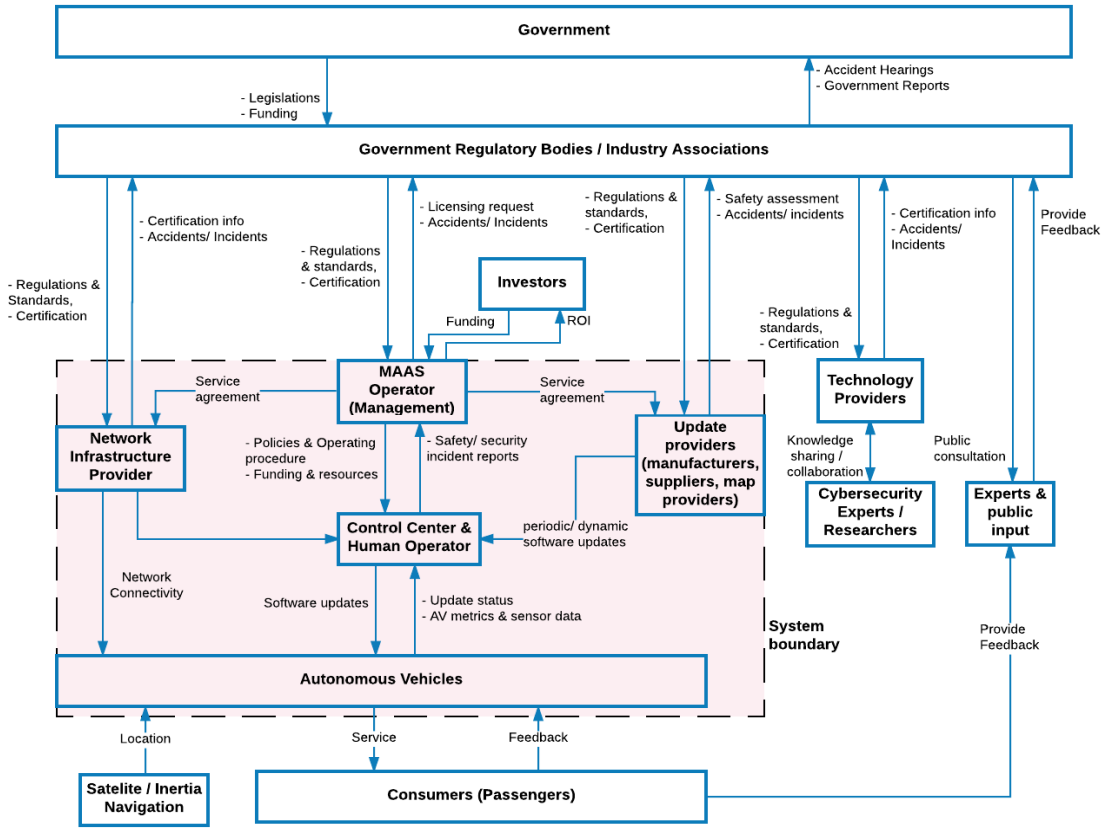

Figure 8. Functional control structure for software OTA updates

Unsafe/ Unsecure Control Actions

The next step is to identify unsafe/ unsecure control actions by assessing control loops within the functional control structure. This analysis is not limited to electro-mechanical components; they can be used to analyze organizational or management components within the control structure. Four types of unsafe control action that can lead to a hazardous outcome are considered, namely:

- A control action required for safety is not provided
- An unsafe / unsecure control action is provided that leads to a hazard
- A potentially safe control action is provided too late, too early, or out of sequence
- A safe control action is stopped too soon or applied too long

Based on the four types of UCAs described above, this stage seeks to identify actions that may cause the system to reach a hazardous state. A total of 15 UCAs were generated by considering each interaction in the functional control. For brevity, only 7 examples will be provided in **Table 2**. Starting from the interactions between the control center and the AVs,

three UCAs were identified. UCA-11 identifies a case where unauthorized updates are provided to the AV, potentially leading to hazards H1 (adversaries take over control of safety-critical features of AV) and H2 (AV operating with unsafe/ unsecure/ outdated software). UCA 10 & 12 identify cases where software updates are not applied to the AVs, or not applied in timely manner, leading to the same hazards H1 and H2.

Table 2. Potentially unsafe/ unsecure control actions

Control Action	Not providing causes hazard	Providing causes hazard	Too early / too late/ wrong order causes hazard	Stopping too soon/ applying too long causes hazard
UCAs between MAAS Operator (Management) and software update provider				
Service agreement	**UCA-1:** Service agreement with network provider not provided before system is operationalized [H3, H4]	Not hazardous	Not applicable	Not applicable
UCAs between MAAS Operator (Management) and Control Center				
Safety/ security policies and operating procedure	**UCA-3:** Safety policies and operating procedure not provided before system is operationalized [H1 – H4]	Not hazardous	Not applicable	Not applicable
UCAs between software update providers and Control Center				
Periodic software updates	**UCA-7:** Software updates not provided by providers when new threats/ vulnerabilities exist [H1, H2]	Not hazardous	**UCA-8:** Software updates provided too late by providers when new threats/ vulnerabilities exist [H1, H2]	Not applicable
UCAs between Control Center and AV				
Periodic/ dynamic software updates	**UCA-10:** Software updates not applied to AVs when threats or vulnerabilities exists in AV [H1, H2]	**UCA-11:** Unauthorized software updated into AVs [H1, H2, H4]	**UCA-12:** Software update not applied to AVs in timely manner when threats or vulnerabilities exists in AV [H1, H2, H4]	Not applicable

For each identified UCA, Safety and Security Constrains (SSCs) were recommended at component-level. **Table 3** shows examples of SSCs. These SSCs are high-level requirements and could serve as input for safety/ security features and requirements as part of the guided design process.

Table 3. Extracts of Safety / Security Constrains

Unsafe / Unsecure Control Actions	Possible Safety / Security Constrains
UCA-1: Service agreement with network provider not provided before system is operationalized [H3, H4]	**SC- 1:** The MAAS operator shall establish service level agreement with network service provider to ensure adequate coverage of network, availability, and protection levels against cyber security threats.
UCA-3: Safety policies and operating procedure not provided before system is operationalized [H1 – H4]	**SC- 2:** The MAAS operator shall translate applicable regulatory requirements and standards to safety policies and operating procedures.

UCA-7: Software updates not provided by providers when new threats/ vulnerabilities exist [H1, H2]	**SC- 7:** The MAAS operator shall establish protocols for periodic or ad-hoc software updates upon detection of vulnerabilities.
UCA-8: Software updates provided too late by providers when new threats/ vulnerabilities exist [H1, H2]	**SC- 3:** The MAAS operator shall establish protocols for timely update of critical software updates that need to be installed on AVs expeditiously
UCA-10: Software updates not applied to AVs when threats or vulnerabilities exists in AV [H1, H2]	**SC- 10:** The MAAS operator shall, by working with associated providers, ensure that software updates are provided to provide fixes for detected vulnerabilities.
UCA-11: Unauthorized software updated into AVs [H1, H2, H4]	**SC- 11:** The MAAS operator shall, by working with relevant parties, prevent unauthorized software from being installed into AVs.
UCA-12: Software update not applied to AVs in timely manner when threats or vulnerabilities exists in AV [H1, H2, H4]	**SC- 12:** The MAAS operator shall ensure timely response to vulnerable software by providing fixes/ patches through pre-emptive or quick recovery approach

The high-level safety/ security constrains derived from the analysis up to stage 2 may be sufficient for some analysis. In STPA Stage 3, we select a few UCAs for further analysis to identify scenarios and causal factors which may cause the UCAs to occur.

Stage 3: Identify causes for unsafe/ unsecure control actions and propose mitigation measures

Stage 3 aims to identify possible scenarios where unsafe/ unsecure control actions may occur. This enables us to map out how unsafe control actions may be triggered, facilitating recommendation of safety and security requirements and improvements to system design, organizations policies, or security governance framework. STPA-Sec provides a method to systematically identify possible causes for each identified UCAs. Using the classification of potential control loop disruptions described in **Figure 9**, we analyzed control loops to identify hazardous scenarios and causal factors that may lead to violation of any safety or security constrains. The diagram includes additional considerations (underlined and bolded) extended from STPA to include additional security analysis. For example, the communication between the main controller and secondary controller also includes unauthorized communications (in additional to missing or wrong communications) when we include cybersecurity considerations in our analysis. The control loop analysis provides heuristics to identify potential disruptions that may cause the system to reach a hazardous or vulnerable state.

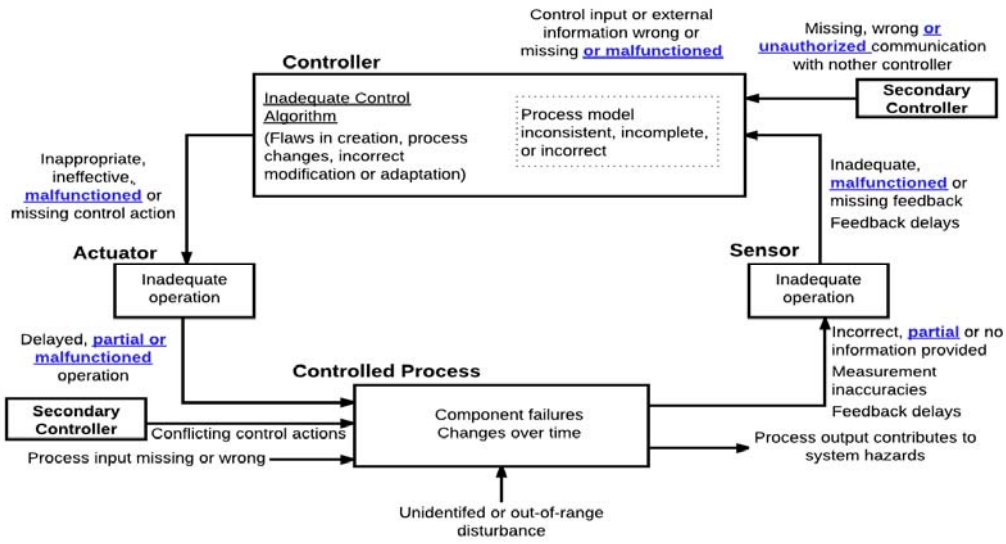

Figure 9. Potential control loop disruptions leading to hazardous states (Adapted from [24])

Interactions between MAAS Control Center and AVs

This example demonstrates the analysis of process models to investigate scenarios and causal factors leading to unsafe/ unsecure interactions between the MAAS control center and the autonomous vehicles (see **Figure 10)**. Starting from the top, software update providers issue software updates to the MAAS control center. To control the distribution of software updates, the control center's process model considers the type of software update (periodic or dynamic), criticality of software update, update mechanisms, and target AVs to be updated. The update process is managed by the OTA software update management system, which distributes approved software updates. Once the updates are downloaded to the AVs, the software packages are installed and the update progress are feedback to the control center.

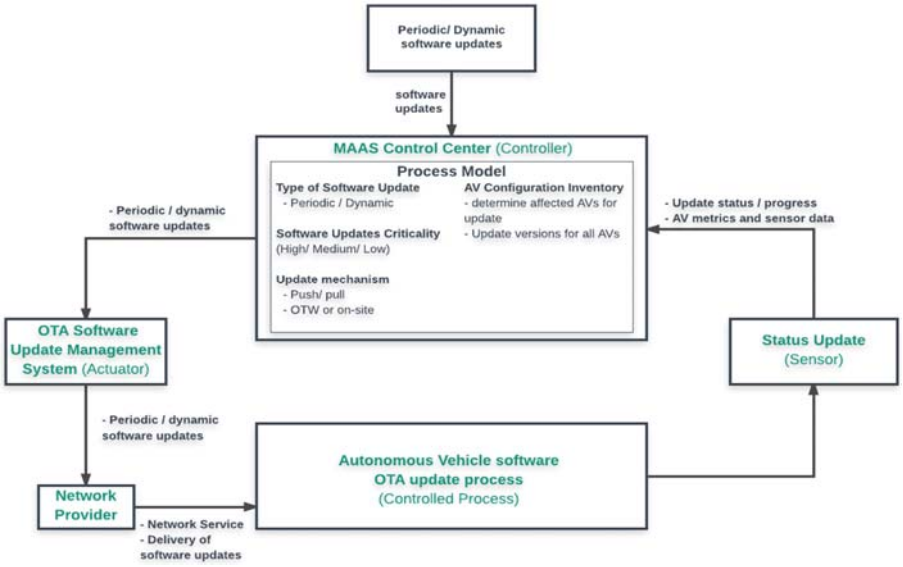

Figure 10. Process model for periodic/ dynamic software updates to AVs

The process model is a crucial step to help us understand why accidents occur. In the stable state, the controllers, actuators, and feedback mechanism ensure the safe and secure operation of software updates process. Using the generic causal factors shown in **Figure 9** as

a guide, potential scenarios for unsafe interactions and their causal factors were identified. The graphical representation of scenarios and causal factors of unsafe interactions is shown in **Figure 11**. Working around the loop, causal factors for each of the components are shown in boxes representing the controller, actuator, control process, and sensors. The detailed scenarios, causal factors, and recommended safety/ security requirements are generated in further details as shown in **Table 4**.

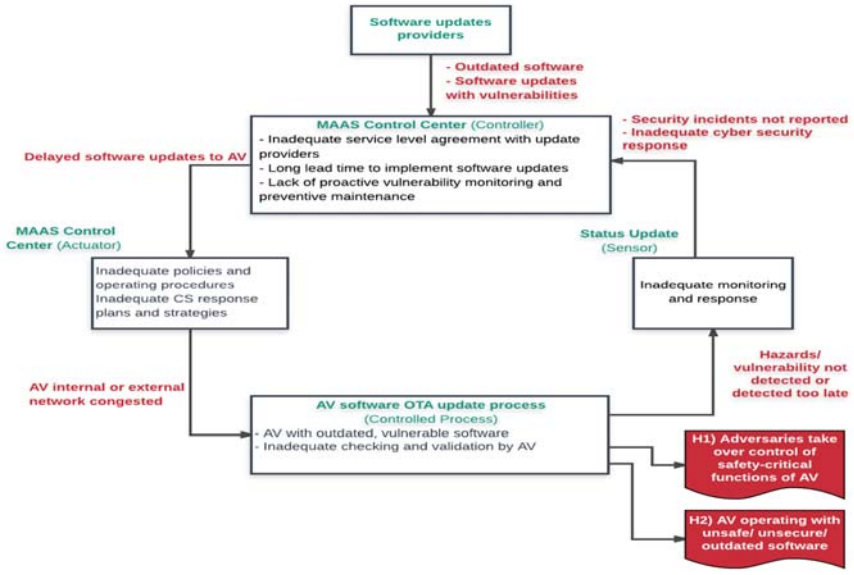

Figure 11. Causal scenarios analysis for periodic/ dynamic software updates to AVs

Table 4. STPA Stage 3 analysis results for periodic/ dynamics software updates to AV

Scenario	Associated Causal Factors	Requirements / Design Features	Allocated to	Rationale
UCA-7: Software updates not provided by providers when new threats/ vulnerabilities exist [H1, H2] **UCA-8: Software updates provided too late by providers when new threats/ vulnerabilities exist [H1, H2]**				
Outdated software Software updates with vulnerabilities	Inadequate service level agreement with update providers	**R-1:** Establish service level agreement with update providers to ensure preventive bugs and vulnerability fixes are included	MAAS Operator (Management team)	Improve organizational cybersecurity plans and strategies
	Long lead time to implement software updates	**R-2:** Ensure pro-active vulnerability monitoring. For example, bug bounty program to invite "ethical hackers" to find security vulnerabilities **R-3:** Dedicated, independent cyber security team actively looking into regular security audit tests and detecting any new threats/ vulnerabilities.	MAAS Operator (Management team)	Improve organizational cybersecurity plans and strategies
	Lack of proactive vulnerability monitoring and security maintenance			
UCA-10: Software updates not applied to AVs when threats or vulnerabilities exists in AV [H1, H2] **UCA-12: Software update not applied to AVs in timely manner when threats or vulnerabilities exists in AV [H1, H2, H4]**				
Delayed software updates to AVs	Inadequate policies and operating procedures	**R-4:** Develop cybersecurity policies and operating procedures to determine update lead time based on different levels criticality.	MAAS Operator (Management team)	Improve organizational cybersecurity plans and strategies

Scenario	Associated Causal Factors	Requirements / Design Features	Allocated to	Rationale
	Inadequate cybersecurity response plans and strategies	**R-5:** Develop cybersecurity response plans and strategies, ensuring all parties involved know their roles and responsibilities in the event of cyber-attack (malicious software updates) **R-6:** Provide a system for response plans to be exercised and lessons learnt incorporated to improve existing plans	MAAS Operator (Management team)	Improve organizational cybersecurity plans and strategies
AV internal or external network over-loaded	Excessive traffic restricting software updates to AV	**R-7:** AV gateway designed to prevent unauthorized traffic	MAAS IT Infra/ security team	Improving Critical Infra Protection
	File size of software updates too large	**R-8:** MAAS operator to work with providers to limit file size of software updates	MAAS Operator	Coordination and cooperation with external providers
	Too many concurrent downloads	**R-9:** MAAS operator to stagger software updates in to minimize network congestion	MAAS Operator	Enhancing internal policies and procedures
	Backdoor attacks within vehicular network (CAN bus)	**R-10:** Segregate networks for safety-critical functions and non-safety-critical functions	AV Manufacturer	Improving AV security design

Additional considerations to STPA-Sec

In the preceding section, we identified causal factors and scenarios by using heuristics in **Figure 9** to identify causal factors and scenarios in which UCAs can occur. While this approach was effective in generating unique and important considerations as part of STPA-Sec stage 3 analysis, other methods may also be used to complement and add additional considerations. In the 2016 STAMP conference [25], Young proposed the use of information lifecycle model to add considerations to the STPA-Sec analysis (see **Figure 12**). To this end, CHASSIS hazards analysis mentioned in the earlier section may be considered. We will compare recommended mitigations from both CHASSIS and STPA-Sec analysis and list additional mitigations generated if we were to incorporate information lifecycle stages to generate additional considerations.

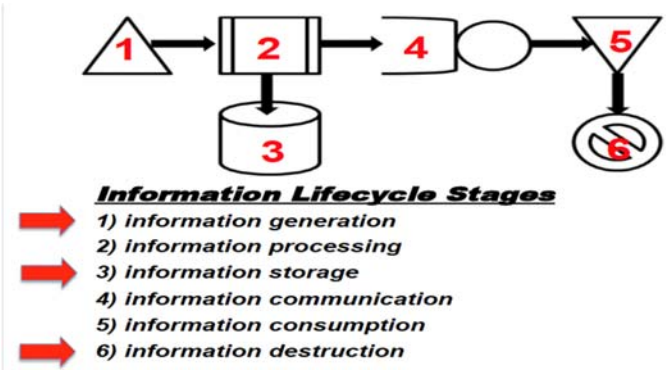

Figure 12. Information lifecycle stages [25]

Summary of STPA-Sec analysis

Overall, STPA-Sec co-analysis of safety and security hazards with the MAAS software OTA update use case demonstrated several potential benefits over traditional methods. Adopting systems thinking approach and analyzing hazards using control theory provides guidance to consider the broader system. Furthermore, the approach facilitates the identification of hazardous state due to unsafe/ unsecure interactions among components, and readily captures causal factors such as managerial decisions, organizational policies, and regulatory landscape arising from socio-technical interactions. The full STPA-Sec analysis (refer to [26]) yielded 44 design requirements. From systems control perspective, these requirements aim to mitigate safety and security risks by controlling feedback loops in order to prevent the system from reaching the hazardous states as defined earlier in this chapter.

The key findings and take-away from the STPA-Sec analysis are summarized as follow:

- STPA is based on a top-down approach; its scope of analysis is bounded by unacceptable accidents/ losses and hazards identified upfront. One key lesson from this study is that even for a relatively narrow system boundary under analysis, the number of causal scenarios and mitigation actions can expand substantially. It is therefore recommended to begin the analysis at higher levels of abstraction, and then go into further details by further in-zooming the functional control structure in subsequent iterations.
- One of the distinguishing feature of STPA approach is the consideration of socio-technical interactions beyond the technical operational aspects of the system. Analyzing the interactions within the whole eco-system can be useful in finding insights on how the external interactions may impact process model of components further down the control structure.
- STPA incorporates heuristics to aid identification of unsafe/ unsecure interactions, as well as causal scenarios in which such interactions may take place. In STPA Stage 2, the four factors in which unsafe control actions can take place (e.g. control action not provided, unsafe/ unsecure control action provided, control action provided too late/ too early/ out of sequence, and control action stopped too soon or applied too long) are to some extend similar to HAZOP guidewords. In STPA Stage 3, in addition to classification of causal factors for identifying possible accident scenarios, it may be useful to apply the Confidentiality, Integrity, and Availability (“CIA”) guidewords as creativity process for identifying cybersecurity causal factors.
- An additional benefit of STPA is the ability to trace mitigation requirements to the hazard(s) that these requirements are intended to mitigate against. Having clear traceability to the intent is important to help developers and testers validate the system, as well as ensure that the mitigation measures are maintained should the system be upgraded or replaced.

Hazard and Risk Analysis Using CHASSIS

Recognizing that safety and security areas are increasingly merged and interconnected in new systems, CHASSIS was proposed as a combined safety and security assessment to minimize resources compared to having separate assessments, as well as support the understanding and resolving potential conflicts between the two areas [14]. Using techniques from both safety and security fields, CHASSIS artefacts are built on the Unified Modelling Language (UML). As part of the safety and security assessment, CHASSIS adopts the Hazard and Operability studies (HAZOP) guidewords as creativity process for generating unsafe/ unsecure actions. HAZOP provides a structured and systematic examination of potential actions by misusers and attackers during the analysis. **Figure 13** provides an overview of process activities in CHASSIS method.

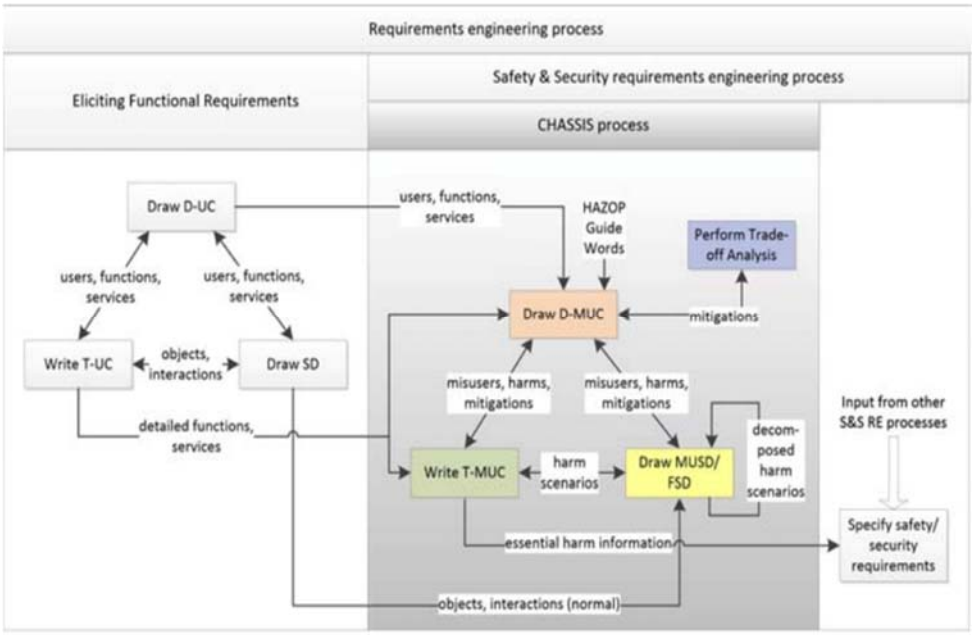

Figure 13. Process activities in CHASSIS [27]

Stage 1: Eliciting functional requirements

Stage 1 focuses on eliciting functional requirements. The functions, users and services within the system are presented in the Diagrammatic-Use Case (D-UC) as shown in **Figure 14.** D-UC showing functions, services, and users involved in the system

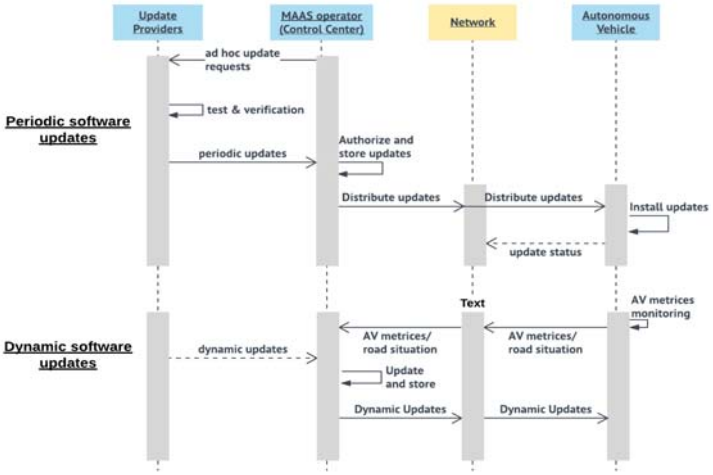

Figure 15. SD showing system interactions for periodic and dynamic software updates

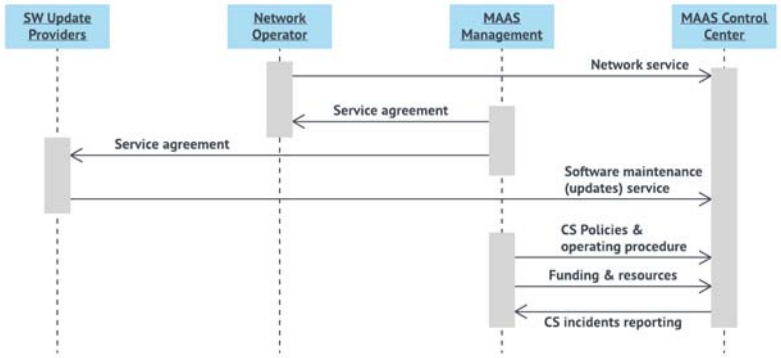

Figure 16. SD showing interactions involving MAAS management

. The D-UC is further extended to represent the interactions and activities in the sequence diagrams. **Figure 15** shows the technical sequence of activities for both periodic and dynamic updates that are disseminated to autonomous vehicles. Error! Reference source not found. shows the managerial and organizational activities, including the establishment of service agreements, generation of internal policies and guidelines, and funding and resources allocated to strengthen cybersecurity measures. Further descriptions of the possible paths of activities are elaborated in Textual use-case (T-UC) tables – see example in **Table 5**.

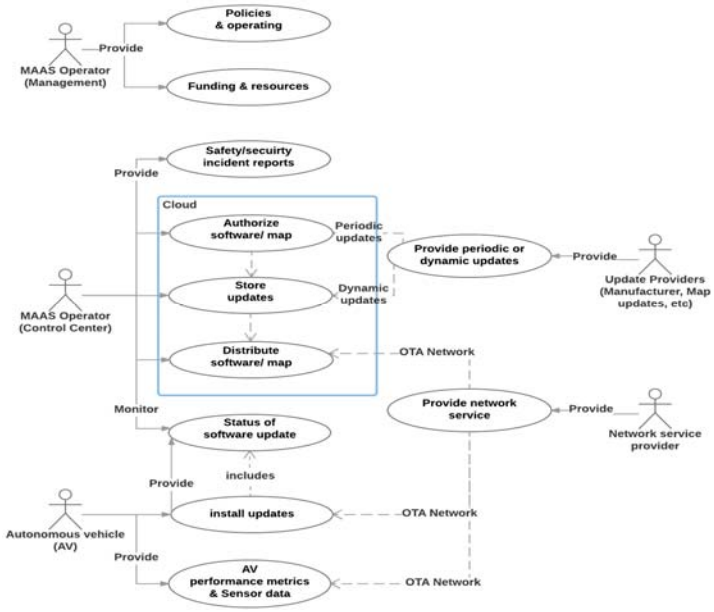

Figure 14. D-UC showing functions, services, and users involved in the system

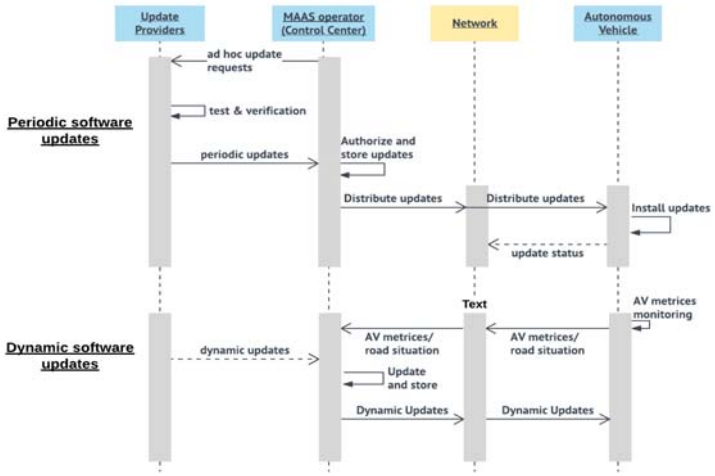

Figure 15. SD showing system interactions for periodic and dynamic software updates

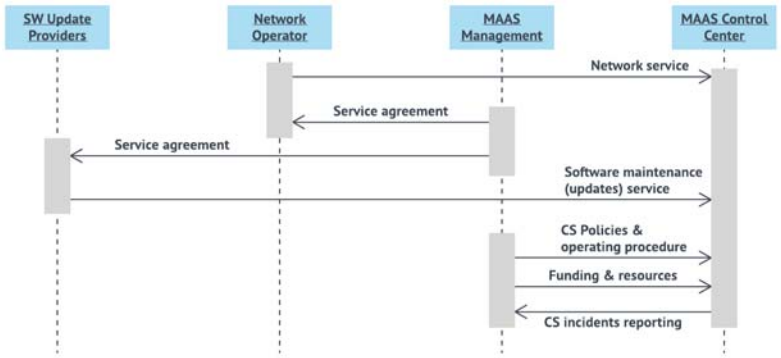

Figure 16. SD showing interactions involving MAAS management

Table 5. Tabular Use Case (T-UC) – Detailed descriptions

Name	MAAS Operator (Management) providing policies and operating procedure
Summary	MAAS management team issues policies and operating procedures related to cybersecurity, ensuring that staff at all level know their roles and responsibilities
Basic Path	BP1: Management decides on applicable security/ safety standards, guidelines, and regulations BP2: Translate into policies and operating procedure for staff to adhere to. BP3: Staff follow policies and operating procedures in their daily work
Alternative Path	AP1: No internal policies and operating procedure; staff follow applicable standards directly
Exception points	ECP1: In BP2, management team ignores applicable security standards and guidelines ECP2: In BP3, staff unsure of CS organizational policies and operating procedure and carry out unsafe/ unsecure actions
Extension Points	ETP1: Management sends staff for additional training to build CS competencies
Triggers	Applicable standards, guidelines, and regulations on safety and security for automotive
Assumptions	Management has policy team whose role is to manage internal organizational policies and procedures
Related business rules	Organizational CS strategy, objectives and missions

Stage 2: Eliciting safety and security requirements

Stage 2 focuses on eliciting and analyzing safety and security requirements. Using existing work in [28][29][30], potential misuse cases (shaded in red) were extended to the D-UC to derive the diagrammatic misuse case (D-MUC) diagrams, as shown in **Figure 17**. Next, scenarios for misuses and harm cases were identified. These misuses and harm cases were identified by associating use cases with guide words from hazard and operability study (HAZOP). This provides a structured way to analyze possible cases. For example, using the guide word “Other than”, the scenario where a flawed update provider may send outdated or wrong updates was identified. Other common guidewords include: Before; After; Early; Late; No or Not; Reverse, etc. To further consider cybersecurity threats, we used the Confidentiality, Integrity, and Availability (CIA) model to identify security-related threats scenarios. For example, by brainstorming possible “Confidentiality” threats, the threat scenario where an attacker may eavesdrop on the network and gather sensitive information was identified.

Misuse scenarios (red boxes) were captured in the Failure Sequence Diagram (FSD), while security threats scenarios (purple solid boxes) were captured in the Misuse Case Sequence Diagrams (MUSD). **Figure 18** shows the combined FSD and MUSD for the misuse case/ threats for periodic software updates originating from the cloud, while **Figure 19** shows the same for interactions involving the MAAS management. By identifying possible bad actors and possible points where misuses can take place, vulnerable points within the system and their downstream impacts can be identified. In the misuse case of flawed management or investors placing more emphasis on areas other than cybersecurity, MUSD in Figure 19 identified possible flaw with inadequate funding and resources allocated to the MAAS Control Center. Inadequate cybersecurity measures can propagate to downstream impacts ranging from the lack of policies and guidelines for protection and response strategies to an increase in likelihood of cyber-attacks (V1 to V6 in Figure 18).

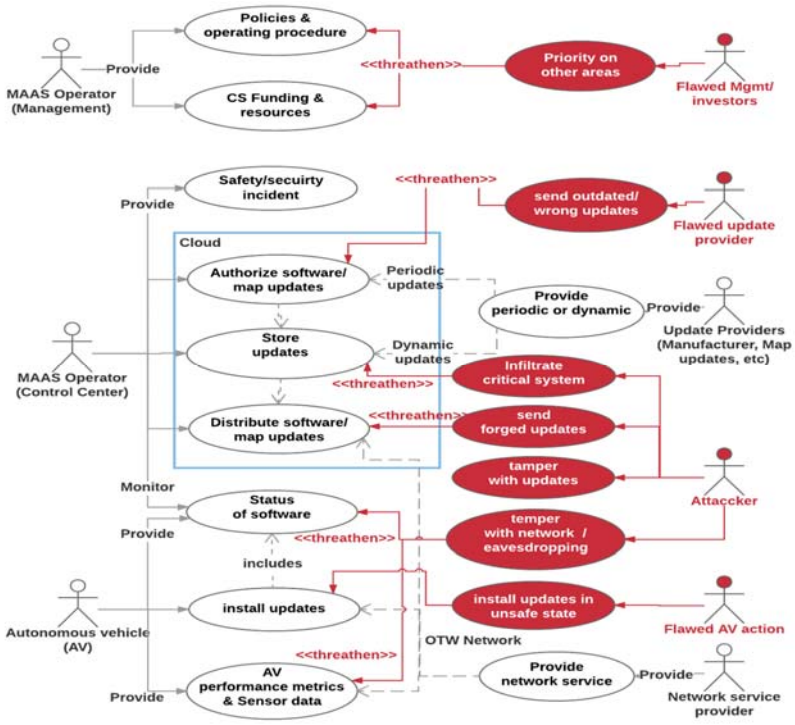

Figure 17. D-MUC for software/ firmware OTA update process

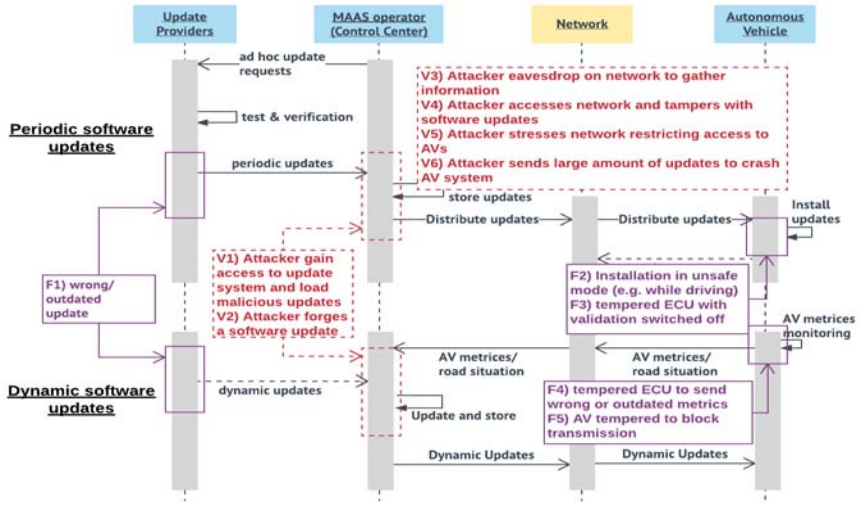

Figure 18. FSD and MUSD for periodic and dynamic software updates

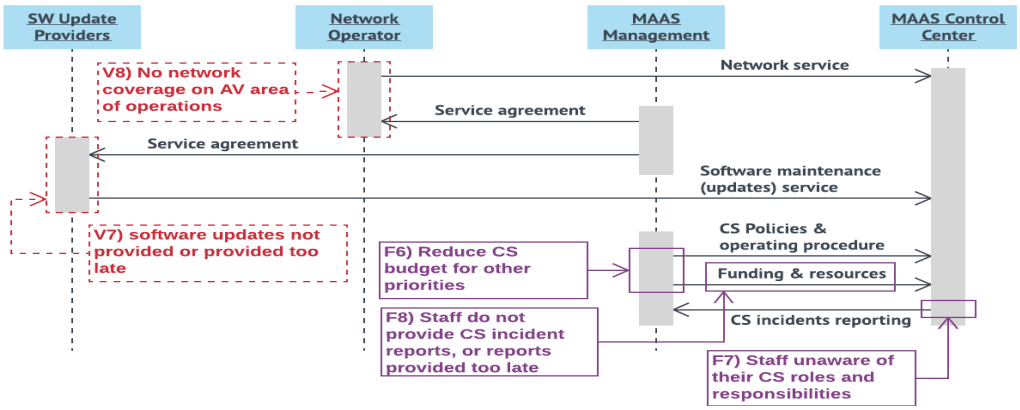

Figure 19. FSD and MUSD for interactions involving MAAS management

System flaws and vulnerabilities from D-MUC and MUSD diagrams are detailed in the Textual-Misuse Case (T-MUC). **Table 6** shows an example T-MUC for the flawed provider sending outdated or wrong updates to the AV manufacturer. The T-MUC further describes the sequence of events leading to the system flaws or vulnerabilities, followed by recommendations to mitigate the risks at various points. Further contextual information, such as assumptions made, pre-conditions, misuser profile, stakeholders involved, and risks are further elaborated in the T-MUC.

Table 6. T-MUC detailed description

Name	F1) Flawed provider send outdated or wrong update to AV manufacturer
Basic Path	BP1: Provider sends outdated or wrong update to MAAS Operator in Control Center BP2: Wrong/ outdated update not detected and is updated on the AVs BP3: Outdated software with vulnerabilities installed on AV, leading to possible hazards
Mitigation points	**MP- 1:** A process or system to prevent wrong or outdated updates from being issued by providers. **MP- 2:** A mechanism or system to detect wrong or outdated updates from being received by MAAS operator

	MP- 3: A system to prevent wrong or outdated update from being installed on AVs.
Assumptions	1) There is a software system to facilitate software updates and distribution
Pre-conditions	1) Wrong or outdated updates can lead to hazards
Misuser profile	1) Mistake/ lapse from faulty provider
Stakeholders, risks	1) Safety of passengers of pedestrians in danger 2) Damage to vehicles or physical properties 3) Both MAAS operator and provider's reputation is at stake

Stage 3: Summary of recommendations

CHASSIS analysis identified a total of eight safety-related flaws and eight security-related system vulnerabilities. The analysis result also found 23 mitigation points to address these system flaws and vulnerabilities. In CHASSIS Stage 3, a summary of the analysis results is presented in HAZOP tables (see example in

Table 7) to record functions that were analyzed, together with parameters and guidewords used to identify potential misuse cases.

Table 7. HAZOP table for software update providers

Function	Guideword	Consequence	Cause	Harm	Recommendation
Provide periodic or dynamic updates	HAZOP "Other than"	F1) Flawed provider sends outdated or wrong update to MAAS operator; the update is distributed to AVs causing it to operate in hazardous state	Mistake/ lapse from flawed provider	AV operating with vulnerable/ unsafe software	**MP- 4:** A process or system to prevent wrong or outdated updates from being issued by providers. **MP- 5:** A mechanism or system to detect wrong or outdated updates from being received by MAAS operator **MP- 6:** A system to prevent wrong or outdated update from being installed on AVs.

Table 8 provides the summary of requirements generated from CHASSIS analysis, categorized into managerial aspects, organizational/ operational aspects, technical (AV design) aspects, and technical (MAAS IT infrastructure) aspects.

Table 8. Summary of requirements and design considerations from CHASSIS analysis.

MAAS Managerial	MAAS Organizational / Operations	Technical (AV Design)	Technical (MAAS IT Infrastructure design)
2 requirements	**7 requirements**	**10 requirements**	**4 requirements**
MP-18: Establish acceptable agreement and mutual understanding on acceptable software support **MP-19**: Establish acceptable agreement and mutual understanding	**Organizational Policies & Processes** **MP- 11:** Develop contingency plans and strategies to recover from potential attacks **MP- 20:** Develop organizational multi-year CS strategy and plan; ensure adequate funding and resources are allocated to meet the plans **MP- 21:** Provide clear CS guidelines, policies, and training to ensure that staff at all levels are familiar with	**Organizational Policies & Processes** **MP- 3:** A system to prevent wrong or outdated update from being installed on AVs. **MP- 6:** Design and develop a mechanism to certify updates are from trusted sources **MP- 8:** AV shall have capability to detect messages that may be altered or are from unidentified sources **MP- 9:** AV shall ensure updates are from authorized and authenticated sources before receiving these updates	**Data-at-rest Protection** **MP- 4:** Ensure Critical IT infrastructure is protected 24/7, with access controls in place **MP- 5:** Ensure network access to internal IT system is adequately protected with authentication and authorization mechanisms **MP- 10:** Design and implement protection of critical

on acceptable network coverage within AV's area of operations	their CS roles and responsibilities **Operations** **MP- 1:** A process or system to prevent wrong or outdated updates from being issued by providers. **MP- 2:** A mechanism or system to detect wrong or outdated updates from being received by MAAS operator **MP- 22:** Include standard operating procedures to update management on CS incidents based on criticality/ severity **MP- 23:** Automated system to alert management on critical CS incidents	**MP- 12:** Design the AV such that it does not allow installation of critical software updates while driving **MP- 13:** Ensure that security feature that prevents software installation in hazardous mode is tamper-proof **MP- 14:** Design and implement authentication and authorization mechanism **MP- 15:** Ensure that the authentication and authorization mechanism is tamper-proof **MP- 16:** Design and implement tamper-proof ECU to prevent unauthorized inject of commands to ECU to send wrong/ inaccurate metrics **MP- 17:** Design and implement tamper-proof communications module to prevent the module from being disabled	infrastructure to block any invalid traffic (e.g. firewalls, strong password, network encryption) **Data-in-transit Protection** **MP- 7:** (1) Ensure security-critical information are sent using secure network protocol; (2) Ensure security-critical information are encrypted before transmission

Comparison between STPA-Sec and CHASSIS

Another key objective of this research is to compare the results from STPA-Sec analysis with another safety/ security hazard analysis method – CHASSIS. **Table 9** compares the number of requirements for each category generated from both STPA-Sec and CHASSIS. Across all categories, it is observed that more requirements were generated from STPA-Sec analysis.

Table 9. Comparison of recommended mitigation measure types between STPA-Sec and CHASSIS

Categories of requirements	STPA-Sec	CHASSIS
Managerial aspects	**7**	**2**
Organizational / Operations aspects	14	7
Technical (AV design) aspects	13	10
Technical (MAAS IT Infrastructure) aspects	14	4
Total requirements	48	23

A qualitative comparison of both hazards analysis methods is provided in **Table 10**. In the previous section, the use of CHASSIS for information lifecycle analysis to add considerations to STPA-Sec analysis was suggested. Based on the comparison of analysis results, there are strengths in CHASSIS analysis that makes it feasible to complement STPA-Sec stage 3 analysis to generate additional considerations. In particular, we found that because CHASSIS focuses on possible activities by misusers and attackers at various stages of information lifecycle, the analysis generated layered defense requirements with specific prevention and detection mechanisms at key components to prevent propagation of vulnerabilities. The information lifecycle analysis (using CHASSIS models) could be conducted in subsequent stages of STPA-Sec to derive additional requirements at components level.

Table 8. Qualitative Comparison of recommended mitigations from STPA-Sec and CHASSIS

Mitigations in both STPA-Sec and CHASSIS	Mitigations in STPA-Sec only	Mitigations in CHASSIS only
Both methods identified mitigations to strengthen control	- Strength in identifying mitigations from socio-technical aspects that	- Strength in identifying layered defense; specific prevention and

and protection at key components under system boundary.	have indirect interactions with technical system (considered investors, standards and regulations, staff training and competency, environmental impacts, etc.). - Significantly more requirements from managerial, organizational policies/ processes from STPA-Sec; more focus on top-down control mechanisms than bottom-up fixes	detection requirements for each component within information chain. - Focused on possible activities by misusers and attackers - Unique requirements include: identification of wrong or outdated software updates due to human error; tamper-proof design for critical features to mitigate attackers' activities

The in-depth comparisons between STPA-Sec and CHASSIS are shown below. Several differences which may affect its effectiveness in specific context are highlighted.

Analysis approach

Both STPA-Sec and CHASSIS encompass co-analysis of safety and security hazards. At its core, STPA-Sec is built on control theory with hazards and vulnerabilities a result of inadequate controls within the system. As a result, the technique enables identification of unsafe interactions even if individual components are working perfectly. On the other hand, CHASSIS model system behavior based on information flow and interactions; it facilitates identification of system failures or vulnerabilities based on activities introduced by misusers or attackers. We also observed that for analysis at high-level abstractions, in-depth knowledge of the system is not a necessity, which makes both STPA-Sec and CHASSIS feasible for teams without strong expertise in the system.

Level of abstraction

Since STPA-Sec is based on STAMP, its underlying concept is to analyze the system taken as a whole, rather than its components taken separately. STPA-Sec is a high-level, top-down approach, focusing on emergent properties that arise from relationships among components. Therefore, the technique is well suited for systems in early stages of development and concept phase where architectural artefacts have not been established. STPA-Sec generates more high-level requirements considering the larger socio-technical aspects of the system. In contrast, CHASSIS can be considered a bottom-up approach building upon functional decomposition of key components use cases and their interactions. To establish the use cases and information sequence flow of the system, some high-level functional requirements and information transactions would be required during analysis. The mitigations generated by CHASSIS are strong in generating layered defense against possible activities by misusers and attackers to prevent the vulnerability/ fault from propagating through the system.

Scope of analysis

STPA-Sec first establishes the high-level control structure encompassing socio-technical interactions with components of the system. The system boundary is then defined to set the focus of analysis on components that are within the team's influence. Next, definition of unacceptable losses/ accidents and hazards also narrow down identification of unsafe control actions to those that attribute to the hazards. Based on using STPA-Sec in this research, it is common to see the scope of analysis can expand, especially in STPA-Sec Stages 2 and 3. Therefore, it is worthwhile to consider starting with a high-level abstraction for the control structure and unacceptable losses/ accidents and then proceed with more in-depth analysis in later stages of STPA-Sec analysis.

One of the key features of CHASSIS is the use of UML to model system interactions using use case/ misuse case diagrams and sequence diagrams. These artefacts may be extended from system design documentations to include system flaws and vulnerabilities that may lead to harm. Furthermore, the diagrammatic UML representations provides a key strength of CHASSIS as these representations of system interactions are intuitive and can easily be conveyed to key stakeholders during discussions. Although CHASSIS have not been designed to consider external organizational and environmental interactions, it is possible to

expand the boundary in subsequent iterations, as demonstrated where the interactions between the management team and control center was expanded to the risk scenario considerations.

Case Analysis of recommendations from STPA-Sec

Next, we evaluate effectiveness of the list of recommendations generated from the STPA-Sec analysis by back testing against a past cyber hack scenario. In a series of hacks starting in 2013, Miller and Valasek demonstrated how potential hackers can gain access to the vehicles over the internet. The experiment, conducted on various car models including the Jeep Cherokee, Toyota Prius, and Ford Escape, demonstrated the ability to remotely control the vehicle's fan, music volume, wipers, and even safety-critical features like the steering wheels, accelerator and brakes [31]. The generic system architecture of the Jeep Cherokee is shown in **Figure 20.** Generic system architecture and features of the Jeep Cherokee

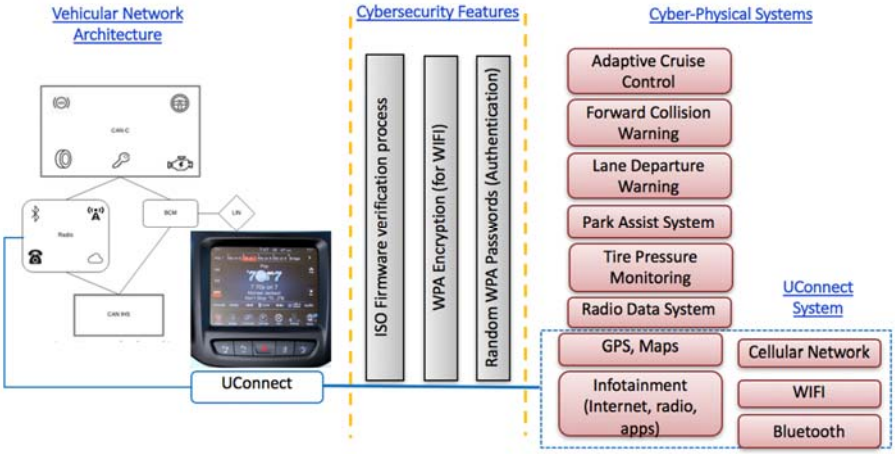

Figure 20. Generic system architecture and features of the Jeep Cherokee

Millier and Valasek identified several vulnerabilities which enabled them to gain access to the vehicle's safety-critical features[2], as depicted in **Figure 2**1 and summarized as follow:

- The researchers identified a micro-controller and software within the UConnect head unit that connects to other components of the vehicle through the vehicle's internal network known as the CAN (Controller Area Network) bus. The CAN bus is a critical infrastructure that enable communications among the vehicle's electronic control units.
- Using this as an entry point, Miller and Velasek planted their code on the firmware of an entertainment system hardware, disabling checks and balances in the vehicle computer units, and enabling them to send commands to the vehicle's CAN bus.
- To access the vehicular network wirelessly using WIFI, the researchers identified that each vehicle's WPA password was generated based on the epoch time (in seconds) from the time the vehicle was manufactured to the first start up. The researchers were able to narrow down to a few dozen combinations and the WPA password used to access the vehicle network can be guessed quite easily.

[2] There has been some confusion as to whether physical access to the vehicle was required. Although physical access to one such vehicle might have been needed to determine the details, the actual cyber attack was conducted remotely without any physical access to that vehicle.

- The UConnect system uses Sprint's 3G network to communicate with other vehicles, and with the vehicle manufacturer for software updates. The researchers found that that it was possible to communicate with other Sprint devices connected anywhere in the country. This network vulnerability allowed the researchers to increase their range of attack by exploiting cellular access into the vehicle.

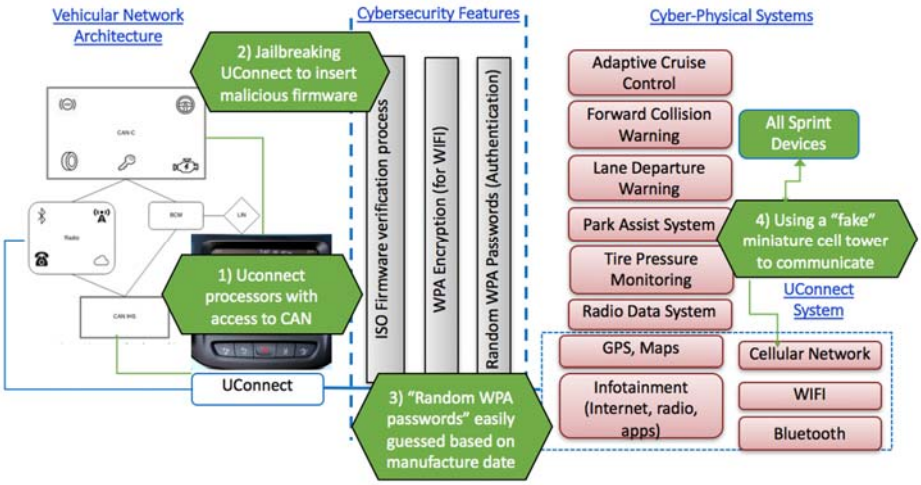

Figure 21. Key vulnerabilities of the Jeep Cherokee

Using the same systems theory based on STAMP, a simplified Causal Analysis using System Theory (CAST) analysis was conducted to analyze deficiencies in the control structure of the Jeep Cherokee case. The goal of this analysis is to identify how system constrains were violated leading to the successful hacks by Miller and Valasek. The accidents/ losses, system hazards and system constrain associated with this incident is shown below.

Associated accidents/ losses:
A1: Financial loss to manufacturer due to recall and rectification of vulnerability
A2: Loss of reputation for manufacturer
A3: Loss of consumer confidence in smart vehicles

Associated system hazard:
H1: Attacker gain access to vehicle to load malicious software [A1, A2, A3]
H2: Attacker gain control of safety-critical functions of vehicle [A1, A2, A3]

Associated system constrain:
SC1: The system control structure must prevent unauthorized software from being loaded to vehicle [H1]
SC2: The system control structure must prevent unauthorized control over safety-critical functions of vehicle [H2]

Figure 22 shows the generic control structure of the vehicle with infotainment controller unit similar to Jeep Cherokee's UConnect feature. The feature allows the human operator (driver) to access and control infotainment features of the vehicles through on-board UConnect Dashboard. The UConnect infotainment system also acts as the visual interface between the driver and vehicular ECUs (such as entertainment system). Vehicular features such as software updates, navigation, telematics, entertainment, and connectivity are available through the vehicle's UConnect feature. The control structure is extended to include the manufacturer, which may receive software update requests following authorization by the

human operator. The manufacturer has the capability to provide software updates to the vehicle over the air.

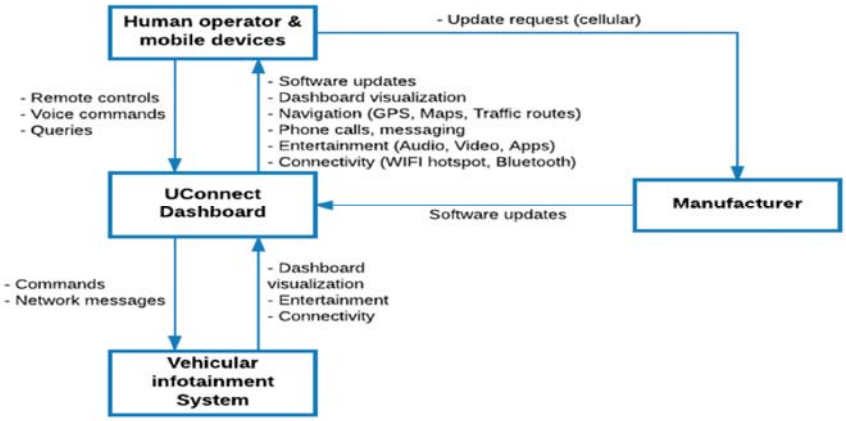

Figure 22. Generic control structure of the Jeep Cherokee system under analysis

Table 11 – 13 provide analysis of safety/ security constrains violated. The analysis highlights key safety responsibilities and constrains violated in each component, together with any emergency and safety equipment present, physical failures and inadequate controls, as well as contextual factors:

- At the physical system (vehicle) level, the unauthorized control of vehicle's safety-critical components results in violation of safety/ security constrain. The driver is also unable to safely over-ride or take-over control of the vehicle. Based on the analysis, the physical failure or inadequate control that led to the violation are due to the Electronic Control Unit (ECU) executing unauthorised command, as well as indirect linkages in ECUs performing safety-critical functions and infotainment functions. Refer to Error! Reference source not found. for details.
- For the in-car controller, the safety/ security constrains violated include the transmission and execution of unauthorised commands, and the unauthorised access to the vehicle's safety-critical features. A number of inadequate controls were observed, ranging from non-encrypted messages, inadequate authorisation and authentication mechanisms, and availability of a back door for attackers to insert malicious codes. Refer to Error! Reference source not found. for details.
- For the vehicle manufacturer, possible safety/ security constrains include inadequate secure development process, cyber-security-related competencies and as well quality assurance processes to ensure cybersecurity risks are mitigated. It is also important to consider contextual factors, such as the supply of parts from different manufacturers, and the competitive industry that calls for new features that may lead to new attack surfaces. Refer to Error! Reference source not found. for details.

Table 9. Analysis of Physical System (Vehicle)

Domain	Analysis
Safety and security responsibilities and constraints violated	- Prevent unauthorized control of vehicle's safety-critical components
Emergency and Safety Equipment (Controls)	- Driver unable to over-ride or take over command of vehicle - Driver may attempt to take over control or switch off the vehicle

Physical Failures and Inadequate Controls	- ECU for safety-critical components execute unauthorized command - ECU for safety-critical components (e.g. accelerator, brakes, steering) and non-safety critical components (infotainment, wipers, etc.) on the same network bus
Contextual Factors	- Driver may completely switch off vehicle but this may be dangerous while the vehicle is driving. - Driver may enable vehicular network (e.g. WIFI, cellular) and expose the vulnerability

Table 10: Analysis of Physical System (in-car controller)

Domain	Analysis
Safety and security responsibilities and constraints violated	- Prevent unauthorized software or command from being sent and executed on vehicle controller - Prevent access to vehicle's safety-critical features
Emergency and Safety Equipment (Controls)	- UConnect system do not have direct features to control safety-critical features of the vehicle.
Physical Failures and Inadequate Controls	- Messages transmitted over the air is not encrypted which allow attackers to interpret messages and plan attacks - Inadequate authorization and authentication allow unauthorized software to be installed in vehicle - Vehicle installed with vulnerable software provide backdoor for attackers to send commands and remotely control safety-features of vehicle
Contextual Factors	- By default, UConnect is designed with features for drivers to remotely control non-safety critical features of vehicle. - Vehicle is accessible from anywhere through the internet which makes it possible for vehicle to launch large-scale attacks remotely - Unsecure interaction between the infotainment system and CAN bus that connects to safety-critical features

Table 11: Analysis of vehicle manufacturer

Domain	Analysis
Safety and security -related responsibilities	- Ensure secure development process, including safety/ security hazards analysis, development, and testing. - Ensure staff are trained in cybersecurity - Ensure vehicles manufactured are secure with cybersecurity risks mitigated
Unsafe decisions and control	- Unsafe interactions of new UConnect feature with existing architecture that shares vehicular network between safety-critical and non-safety critical features - Procure parts/ components with vulnerabilities - Inadequate training or resources on cybersecurity
Process/ mental flaws	- Manufacturer assumed UConnect can only access infotainment features and it is not possible to remotely access safety-critical features - Use of legacy components and parts
Context in which decisions were made	- Highly competitive industry which may cause manufacturers to develop new features to attract buyers - May not be aware of security vulnerabilities for parts/ components procured from suppliers. - Automotive industry may be new to cybersecurity risks and the team may not have adequate competencies

Besides identifying causal factors of safety/ security incidents, CAST advocates the understanding of contextual factors to help us consider external conditions and systemic factors that result in the inadequate controls or unsafe decisions. For instance, based on the analysis of physical failures and inadequate controls, one may be tempted to conclude that the root causes for vulnerabilities were due to poor engineering design which allowed the attackers to access the vehicle's CAN bus indirectly through the infotainment system. Another possible root cause could be due to inadequate transmission and data protection to prevent the system from receiving and executing unauthorized commands. However, based on the report, there were some levels of cybersecurity protections and the researchers were only able to perform the hacks after extensive research. To look beyond human flaws and design errors, CAST also considers other factors such as emergency and safety equipment controls that may not have worked during the accident, as well as contextual factors leading to how existing controls were not effective in preventing the accident.

The in-depth analysis identified several additional unsafe interactions such as:

- Inadequate control for driver to over-ride or take over control of vehicle when vehicle is compromised by attacker
- Unsafe/ unsecure interactions between remote-accessible infotainment system and CAN bus connected to safety-critical features can lead to security vulnerability
- Lack of feedback to alert the driver or manufacturer when the vehicle's safety and security features were compromised. Possible points of compromise include jailbreaking the UConnect console to enable unauthorized software updates, installation of unauthorized software updates, or suspicious access into the system to alter safety-critical features of the vehicle.

Additionally, by extending the control structure to higher hierarchical levels, more in-depth analysis was conducted to analyze contributions of managerial and organizational factors to the vehicle's vulnerabilities:

- Awareness and technical competencies of staff in vehicular cybersecurity
- Effect of competition and time to market on security test adequacy
- Effect of the lack of standards and regulations on cybersecurity in the automotive industry
- Adequacy of cybersecurity activities (design, build, test) in the automotive development approach
- Adequacy of training and resources allocated to cybersecurity efforts in the organization

The unsafe and unsecure interactions identified from the research are represented in the updated control structure found in **Figure 23** (boxes and lines in red represent the unsafe/ unsecure interactions). Evidently, some inadequate controls and feedback were identified which enabled the researchers to identify vulnerabilities and gain access into the system. Some critical control loops not present in the initial control structure (**Figure 22**) were identified. For instance, vulnerabilities in the cellular network access allowed the researchers to identify IOT devices connected within the Sprint network. While it may not be clear whether the network service provider is responsible for securing communications to IOT devices connected to their network, it is a clear indication on the importance to consider network service providers in security risk analysis and to ensure that cybersecurity responsibilities of key parties in the supply chain are clearly defined and executed.

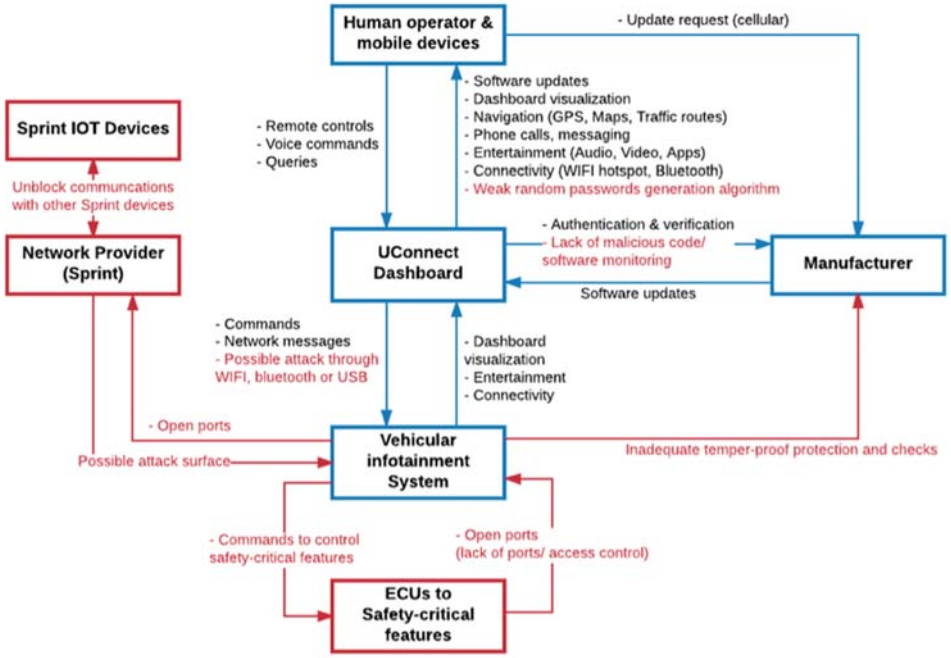

Figure 23. Revised control structure representing unsafe/ unsecure interactions identified from research

Table 12 summarizes the inadequate controls and unsafe decisions identified from this simplified CAST analysis. Requirements generated from both CHASSIS and STPA-Sec analysis (discussed in the earlier sections) that may mitigate these inadequate controls and unsafe decisions are included in the table. We observe that all identified inadequate controls and unsafe decisions are mapped to at least one high-level mitigation measure from STPA-Sec. In contrast, mitigation measures from CHASSIS analysis did not cover some of the identified inadequate controls and unsafe decisions identified in this CAST analysis. It is worthwhile to note that the cases under analysis in STPA-Sec method and CAST method are slightly different – the STPA-Sec method is applied on analyzing hazards and mitigations for OTA software update on autonomous vehicle case, while the CAST method is applied on analyzing causes for a cyber hack on a connected vehicle platform (UConnect). Despite the differences in cases under analysis, the high-level analysis approach in STPA-Sec is able to generate broader requirements that cover both cases. Although this does not guarantee that this cyber hack would not have been possible if the manufacturer had followed all requirements generated from the STPA-Sec analysis, it would be able to address some of the loopholes identified by the researchers, making the hacks more difficult.

Table 12: Inadequate controls/ unsafe decisions identified from CAST analysis and mapping to mitigations from STPA-Sec/ CHASSIS analysis

Inadequate controls/ unsafe decisions	Example of mitigations from STPA-Sec	Examples of mitigations from CHASSIS
ECU for safety-critical components execute unauthorized command	**R-7:** AV gateway designed to prevent unauthorized traffic **R-21:** AVs to check for certified updates before processing these updates **R-26:** In-build intrusion detection system **R-32:** build in tamper-proof design for critical functions of AV	**MP- 6:** Design and develop a mechanism to certify updates are from trusted sources **MP- 8:** AV shall have capability to detect messages that may be altered or are from unidentified sources **MP- 9:** AV shall ensure updates are from authorized and authenticated

Inadequate controls/ unsafe decisions	Example of mitigations from STPA-Sec	Examples of mitigations from CHASSIS
		sources before receiving these updates
ECU for safety-critical components (e.g. accelerator, brakes, steering) and non-safety critical components (infotainment, wipers, etc.) on the same network bus	**R-10:** Segregate networks for safety-critical functions and non-safety-critical functions **R-33:** Build protection mechanism to prevent unauthorized traffic from accessing AV internal traffic	Nil
Messages transmitted over the air not encrypted which allow attackers to interpret messages and plan attacks	**R-15:** Protect communications channel, using secure transport protocol and encryption techniques whenever possible	**MP-7:** (1) Ensure security-critical information are sent using secure network protocol; (2) Ensure security-critical information are encrypted before transmission
Inadequate authorization and authentication allow unauthorized software to be installed in vehicle ECU	**R-23:** Enforce strong authentication and authorization mechanisms to ensure validity of commands/ software **R-20:** Create a certificate authority, with all updates submitted for certification before they can be accepted by AV	**MP- 14:** Design and implement authentication and authorization mechanism **MP- 15:** Ensure that the authentication and authorization mechanism is tamper-proof **MP- 6:** Design and implement tamper-proof ECU to prevent unauthorized inject of commands to ECU to send wrong/ inaccurate metrics
Vehicle installed with vulnerable software provide backdoor for attackers to send commands and remotely control safety-features of vehicle	**R-46:** Provide anomalies detection and analysis tool to detect potential attacks. **R-25:** Send alerts to control stations when unauthorized modifications are detected	Nil
Unsafe interactions of new UConnect feature with existing architecture that shares vehicular network between safety-critical and non-safety critical features	**R-22:** Enforce secure software development lifecycle (SDLC) and conduct audits/ checks to ensure development teams follow them	Nil
Inadequate training or resources on cybersecurity	**R-37:** Build/ strengthen CS technical competencies in organization **R-43:** Ensure AV manufacturer has known track records for safety and security **R-39:** Ensure that staffs at all levels are familiar with their CS roles and responsibilities **R-42:** Translate applicable standards and regulatory guidelines into actionable tasks for the organization	**MP-21:** Provide clear CS guidelines, policies, and training to ensure that staff at all levels are familiar with their CS roles and responsibilities **MP-22:** Include standard operating procedures to update management on CS incidents based on criticality/ severity

Conclusion

This research presented the application of a new safety and security co-analysis based on STAMP theory. Premised on viewing safety as control issues (rather than reliability problem), STAMP has been applied on a wide variety of applications to find inadequate controls that led to an accident or loss (CAST) and to find new requirements to prevent accidents and losses (STPA). STPA-Sec is an extension to STPA to include security analysis. The results were compared with another promising safety and security co-analysis method, CHASSIS, which is built on existing safety and security concepts and information flows to represent both use cases and misuse cases.

Applying STPA-Sec on the MAAS OTA software update case, the analysis identified several unique and important causal factors and mitigation requirements not identified under CHASSIS analysis. The key strengths of STPA-Sec include the ability to identify unsafe interactions among components that may lead the system to hazardous state, and considerations of socio-technical interactions beyond the technical aspects of the system. Overall, STPA-Sec generated more requirements that have greater impact on addressing control weaknesses in the system. The research demonstrated STPA-Sec's ability to identify control flaws which may be missed out in traditional hazards analysis methods. Some strengths in CHASSIS analysis were identified and it was proposed to complement CHASSIS for information lifecycle to generate additional considerations analysis in STPA-Sec stage 3. Several differences between STPA-Sec and CHASSIS which may affect their effectiveness in specific contexts were highlighted.

Finally, the mitigation requirements from both methods were evaluated by back testing against a past cyber hack scenario involving the remote hack experiment of the Jeep Cherokee. Using a simplified CAST analysis, inadequate controls and unsafe decisions from the cyber hack scenario were generated, and mitigation requirements from STPA-Sec analysis were assessed to evaluate the adequacy in addressing vulnerabilities identified in the CAST analysis. It was observed that STPA-Sec generated requirements that were mapped to all inadequate controls and unsafe decisions identified in the CAST analysis, although the cases under analysis and the control structures in both cases were slightly different. In contrast, mitigation requirements from CHASSIS analysis did not address some of the inadequate controls and unsafe decisions identified in the Jeep Cherokee case.

Overall, this research, along with the new STPA-Sec method should serve to encourage further exploration of STAMP-based approach in cybersecurity, particularly in the automotive domain. Although inherent cybersecurity risks will continue to exist with new features and technologies introduced to vehicles, it is possible to mitigate most of the risks by adopting a holistic, top-down approach and considering socio-technical interactions within the system.

Acknowledgements

This material is based, in part, upon work supported by Cybersecurity at MIT Sloan (CAMS): *The Interdisciplinary Consortium for Improving Critical Infrastructure Cybersecurity*, $(IC)^3$.

Bibliography

[1] Tasha Keeney, "Mobility-as-a-Service: Why Self-Driving Cars Could Change Everything." ARK Invest, 25-Oct-2017.

[2] S. Weilun, "nuTonomy driverless-car accident due to 'extremely rare' software glitches; one-north trial resumes," *The Business Times*. [Online]. Available:

http://www.businesstimes.com.sg/transport/nutonomy-driverless-car-accident-due-to-extremely-rare-software-glitches-one-north-trial.
[3] J. Golson, “Tesla and Mobileye disagree on lack of emergency braking in deadly Autopilot crash,” *The Verge*, 01-Jul-2016. [Online]. Available: https://www.theverge.com/2016/7/1/12085218/tesla-autopilot-crash-investigation-radar-automatic-emergency-braking.
[4] “Controlling vehicle features of Nissan LEAFs across the globe via vulnerable APIs,” *Troy Hunt*, 24-Feb-2016. [Online]. Available: https://www.troyhunt.com/controlling-vehicle-features-of-nissan/.
[5] “Hackers Remotely Kill a Jeep on the Highway—With Me in It,” *WIRED*. [Online]. Available: https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/.
[6] “ASQ: Automotive Security: Challenges, Standards, and Solutions.” ASQ, 2016.
[7] N. G. Leveson, *Engineering a safer world: systems thinking applied to safety*. Cambridge, Mass.: The MIT Press, 2012.
[8] “J1739: Potential Failure Mode and Effects Analysis in Design (Design FMEA) and Potential Failure Mode and Effects Analysis in Manufacturing and Assembly Processes (Process FMEA) and Effects Analysis for Machinery (Machinery FMEA) - SAE International.” [Online]. Available: http://standards.sae.org/j1739_200208/. [Accessed: 01-Sep-2017].
[9] C. A. Ericson (II.), *Fault Tree Analysis Primer*. CreateSpace Independent Publishing Platform, 2011.
[10] A. Banerjee, K. K. Venkatasubramanian, T. Mukherjee, and S. K. S. Gupta, “Ensuring Safety, Security, and Sustainability of Mission-Critical Cyber #x2013;Physical Systems,” *Proc. IEEE*, vol. 100, no. 1, pp. 283–299, Jan. 2012.
[11] D. Schneider, E. Armengaud, and E. Schoitsch, “Towards Trust Assurance and Certification in Cyber-Physical Systems,” in *Computer Safety, Reliability, and Security*, 2014, pp. 180–191.
[12] 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)*, 2015, pp. 621–624.
[13] “Security Application of Failure Mode and Effect Analysis (FMEA) | SpringerLink.” [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-10506-2_21. [Accessed: 19-Aug-2017].
[14] C. Raspotnig, P. Karpati, and V. Katta, “A Combined Process for Elicitation and Analysis of Safety and Security Requirements,” in *Enterprise, Business-Process and Information Systems Modeling*, Springer, Berlin, Heidelberg, 2012, pp. 347–361.
[15] D. Dominic, S. Chhawri, R. M. Eustice, D. Ma, and A. Weimerskirch, “Risk Assessment for Cooperative Automated Driving,” in *Proceedings of the 2Nd ACM Workshop on Cyber-Physical Systems Security and Privacy*, New York, NY, USA, 2016, pp. 47–58.
[16] Rodrigo Sotomayor Martinez, “System Theoretic Process Analysis of Electric Power Steering for Automotive Applications.” Jun-2015.
[17] Nathaniel Arthur Peper, “Systems Thinking Applied to Automation and Workplace Safety.” Jun-2017.
[18] Cody Harrison Fleming and Nancy Leveson, “Improving Hazard Analysis and Certification of Integrated Modular Avionics,” *J. Aerosp. Inf. Syst.*, vol. 11, no. 6, Jun. 2014.
[19] Todd Pawlicki, Aubrey Samost, Derek Brown, Ryan Manger, Gwe-Ya Kim, and Nancy Leveson, “Application of systems and control theory-based hazard analysis to radiation oncology,” *J. Med. Phys.*, 2016.

[20] William Young and Nancy Leveson, “An Integrated Approach to Safety and Security Based on Systems Theory,” *Commun. ACM*, no. 57, pp. 31–35, Feb. 2014.
[21] Hamid Salim; Stuart Madnick, “Cyber Safety: A Systems Theory Approach to Managing Cyber Security Risks – Applied to TJX Cyber Attack,” *Work. Pap. CISL 2016-09*, Aug. 2016.
[22] A. Nourian and S. Madnick, “A Systems Theoretic Approach to the Security Threats in Cyber Physical Systems Applied to Stuxnet,” *Work. Pap. CISL 2014-13*, Sep. 2014.
[23] Arash Nourian ; Stuart Madnick, “A Systems Theoretic Approach to the Security Threats in Cyber Physical Systems Applied to Stuxnet,” *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 1, pp. 2–13, Feb. 2018.
[24] William Young and Nancy Leveson, “Systems thinking for safety and security,” *Proc. 29th Annu. Comput. Secur. Appl. Conf. ACSAC 13 ACM*, 2013.
[25] William Young Jr, “Understanding STPA-Sec Through a Simple Roller Coaster Example.” 2016 STAMP Conference Boston, MA, 23-Mar-2016.
[26] Chee Wei, Lee, “A System Theoretic Approach to Cybersecurity Risks Analysis of Passenger Autonomous Vehicles.” Feb-2018.
[27] V. Katta, C. Raspotnig, P. Karpati, and T. Stålhane, *Investigating fulfilment of traceability requirements in a combined process for safety and security assessments*, vol. 6. 2015.
[28] S. Checkoway *et al.*, “Comprehensive Experimental Analyses of Automotive Attack Surfaces,” in *Proceedings of the 20th USENIX Conference on Security*, Berkeley, CA, USA, 2011, pp. 6–6.
[29] D. K. Nilsson, L. Sun, and T. Nakajima, “A Framework for Self-Verification of Firmware Updates over the Air in Vehicle ECUs,” in *2008 IEEE Globecom Workshops*, 2008, pp. 1–5.
[30] M. S. Idrees, H. Schweppe, Y. Roudier, M. Wolf, D. Scheuermann, and O. Henniger, “Secure Automotive On-Board Protocols: A Case of Over-the-Air Firmware Updates,” in *Communication Technologies for Vehicles*, 2011, pp. 224–238.
[31] Chris Valasek; Charlie Millier, “Remote Exploitation of an Unaltered Passenger Vehicle.” 2015.