Safe and Secure Automotive Over-The-Air Updates

Thomas Chowdhury[1(✉)], Eric Lesiuta[1], Kerianne Rikley[1], Chung-Wei Lin[2], Eunsuk Kang[2], BaekGyu Kim[2], Shinichi Shiraishi[3], Mark Lawford[1], and Alan Wassyng[1]

[1] McMaster Centre for Software Certification, Department of Computing and Software, McMaster University, Hamilton, ON, Canada
{chowdt2,lesiutej,rikleykn,lawford,wassyng}@mcmaster.ca
[2] Systems & Software Division, Toyota InfoTechnology Center U.S.A. Inc, Mountain View, CA, USA {cwlin,ekang,bkim}@us.toyota-itc.com
[3] Software Systems Group, System Architecture Research Division, Toyota InfoTechnology Center Co., Ltd. {sshiraishi}@jp.toyota-itc.com

Abstract. Over-the-air updates have been used for years in the software industry, allowing bug fixes and enhancements to desktop, laptop, and mobile operating systems and applications. Automotive vehicles now depend on software to the extent that manufacturers are turning to over-the-air updates for critical vehicle functionality. History shows that our software systems are most vulnerable to lapses in safety and dependability when they undergo change, and performing an update over a communication channel adds a significant security concern. This paper presents our ideas on assuring integrated safety and security of over-the-air updates through assurance case templates that comply with both *ISO 26262* (functional safety) and *SAE J3061* (cyber-security). Wisely, the authors of *SAE J3061* structured the guidebook so that it meshes well with *ISO 26262*, and we have been able to use principles we developed for deriving an assurance case template from *ISO 26262*, to help include compliance with *SAE J3061* in the template. The paper also demonstrates how a specialization of the template helps guide us to pre-emptively mitigate against potential vulnerabilities in over-the-air update implementations.

1 Introduction

The original motivation for over-the-air (OTA) updates to automotive software seems to have been a realization that customers view a trip to the dealership to install a software patch, as an avoidable waste of their time. This is true even when the patch introduces a new feature that they are pleased to install. An update can take place without the presence of the owner. Whether the update is installed automatically or needs approval before driving depends on the criticality of the update. For example, if the update is for parts of the infotainment system, perhaps it can be installed automatically. If the update is for a critical component of the vehicle then it may be necessary to have driver approval. In

all cases, the update will be installed when the car is at home and is stopped in park mode. In addition, original equipment manufacturers (OEMs) hope that OTA Updates will be a lot more cost effective than paying dealerships to install the updates.

However, with the implementation of OTA firmware updates come new entry points for hackers to tamper with a vehicle's software. Not only do we introduce the potential for hacking, but we also remove a trained technician from the process. These trained professionals help validate that the installation of the new firmware is successful, and ensure that there are no safety hazards resulting from the update. For example, even a simple update to an infotainment system caused cycles of rebooting the heads-up display, accompanied by distracting bright purple flashes, thus resulting in a serious safety concern [4].

It is important to note that we are primarily interested in the final safety of the vehicle. To this end we have to consider the safety aspects of OTA Updates, independent of security concerns, as well as the effect of security issues on vehicle safety – and even the adverse effect of safety mitigation on security. Most current research seems to be heavily focused on security, as though it is an end in itself, although we are starting to see significant work on the interplay between safety and security.

The primary contribution of the paper is the development of an *Assurance Case Template (ACT)* that applies to OTA Updates. Our template complies with both *ISO 26262* [10] and *SAE J3061* [24], and applies in general to functional safety (we have not always prefixed safety by "functional", but that is the focus of this paper) and cybersecurity of the "connected car". An important contribution is the demonstration that the template for OTA Updates can be used to guide development (not just to document assurance after/during development) in a way that helps to avoid/mitigate OTA Update vulnerabilities. The scope of cybersecurity in this paper is limited to protecting the download of the updates. We believe that not protecting such downloads is equivalent to a systematic design fault, and is thus of importance to functional safety. Cybersecurity also deals with financial loss, personal identity theft, data loss, etc. These concerns do not have immediate functional safety implications and are thus not considered in this paper.

This paper is organized as follow. A brief introduction to concepts discussed in the paper is provided in Section 2. This includes an introduction to relevant standards, assurance cases, and ACTs. This is followed by a brief discussion on relevant literature in Section 3. Section 4 is the heart of the paper. It presents an overview of the methodology we used to arrive at an ACT that assures both safety and security for vehicles that may be maintained using OTA Updates. This section extends earlier work [7], in which we developed and used principles for transforming clauses in *ISO 26262* into claims and evidence in an ACT. We have now used those same principles applied to *SAE J3061*, to develop an ACT that integrates safety and security for the connected car. The section includes a very brief description of threats and threat analysis that we need to include OTA-specific assurance. We added this OTA assurance to the template, based

on an open source OTA Update design, Uptane [13, 15]. In Section 5 we show how this template can be used by examining a potential vulnerability in an implementation of Uptane, that was discovered by instantiation of the ACT. Finally, we present our conclusions and future work in Section 6.

2 Preliminaries

2.1 Relevant Standards

There are two standards that are specifically relevant to this work. The first is *ISO 26262* which has become the de facto functional safety standard for electric and software components in automotive vehicles. The second is a newer "standard", *SAE J3061*, which is an SAE guidebook specifically targeting automotive security. It is intended as a companion standard to *ISO 26262*, and has been organized to mesh well with *ISO 26262*, but its written structure differs significantly from *ISO 26262*. There is an unpublished standard *ISO/SAE 21434* [11] for cybersecurity of automotive vehicles. The standard defines requirements for cybersecurity risk management for road vehicles throughout the development process [3]. The standard is currently under development.

2.2 Assurance Cases

An assurance case is a living document assuring a system's critical properties. According to Bloomfield et al., "An assurance case is a documented body of evidence that provides a convincing and valid argument that a specified set of critical claims about a system's properties are adequately justified for a given application in a given environment" [5]. An assurance case starts with a top-level claim which is decomposed into sub-claims supported by other sub-claims or evidence. Each (sub-)claim must be supported by its sub-claims and/or evidence. The argument (reasoning) should be explicit. Other artifacts are used in the assurance case to provide *context*, *assumptions*, *justification*, etc. There are various notations for documenting assurance cases, the most popular one currently being *Goal Structuring Notation (GSN)* developed by Kelly [14]. We have used a GSN tool to draw the assurance cases used in this paper.

2.3 Assurance Case Templates

An ACT provides the structure for a family of assurance cases for a particular product-line. Given a specific product in that product line, one instantiates the ACT to create a complete assurance case for that product. The template consists of optional argument paths corresponding to various features of different products from a specific product-line. A major benefit of templates is that they are developed before the systems/products are built, and thus an evidence node in the template contains a guideline for the specific evidence required to support a sub-claim, along with acceptance criteria for that evidence [29]. The original

motivation for a product-specific ACT was that if every assurance case uses a unique structure and argument, regulators will be overwhelmed by the task of evaluating these assurance cases [30]. Another motivating factor was the work by Graydon, Knight and Strunk on Assurance Based Development [9].

Making these templates specific to a product line may yield the following additional benefits:

- Facilitation of incremental certification;
- Robustness with respect to likely changes (reminiscent of information hiding);
- Playing the role of a safety plan with regard to what needs to be produced, for example:
  - Directing developers as to what evidence should be provided to support specific sub-claims, as well as acceptance criteria for that evidence;
  - Structured using arguments built by people with appropriate expertise;
  - Avoiding confirmation bias, due to the template being constructed before development begins; and
  - Providing a publicly accessible example of an assurance case argument and structure.

A skeleton for the claims, sub-claims and evidence in an ACT is shown in Figure 1.

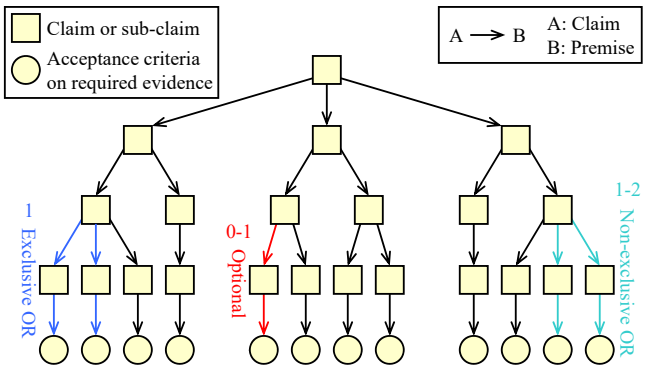

Fig. 1. Basic Structure of an ACT [7]

2.4 Principles for Developing the ACT

The starting point in developing the ACT is the decision to make it compliant with *ISO 26262* and *SAE J3061*. We previously developed and published principles for constructing an ACT from a standard such as *ISO 26262* [7]. The 10 principles are shown in Table 1. We originally used these principles to build an ACT that assures safety and is compliant with *ISO 26262*. We have now used these principles to include security in the ACT, compliant with *SAE J3061*. Although *SAE J3061* is not written in the style of *ISO 26262*, we did manage

Table 1. Principles for Developing an ACT from a Standard

Id	Principle	Description
1	Model the Standard	Understand terms, relationships & requirements
2	Model System Variability	Model detailed enough to guide development
3	Flip-It	Reverse process flow for claim dependency
4	Conjunctive Claims	Multiple claims support a parent claim
5	Optional Pattern	Standard defined alternative processes
6	Evidence Acceptance Criteria	Standard specifies attributes/characteristics
7	Evidence Classification	Determines type of evidence required
8	Completeness	Arguments that depend on completeness
9	Argument options	Optional paths motivated by alternative arguments
10	Feature options	Optional paths motivated by alternative features

to apply those principles sufficiently well to help in the derivation of the new ACT. We supplemented compliance with the standards with our knowledge of safety and software engineering principles. More detail about this process is in Section 4.2.

3 Related Work

Various threat analysis methods are described in [17–19,25,31]. OTA Update specific security is discussed in [13,15,26]. In particular, the *Uptane Project* [13,15] defines an open source software security system with a flexible design, allowing it to be adapted easily to various systems. The Uptane project presents a comprehensive look at common types of attacks that an unsecured vehicle will be vulnerable to, specifically when updated remotely. The attacks described in [13] are: Read Attacks, Replay Attacks, Denial-of-Service (DoS) Attacks (including Drop Attacks, Slow Retrieval Attacks, Flood Attacks, Freeze Attacks), Rollback Attacks, Modify Attacks (including Partial Bundle Attacks, Mixed Bundle Attacks, Mix-and-match Attacks), Spoof Attacks and Control Attacks.

Long-term, we believe that the best way of integrating safety and security is to use an integrated hazard/threat analysis and risk management. Implementing safety and security requirements derived separately from independent hazard analysis and threat analysis may lead to conflicting requirements which result in new hazards and/or vulnerabilities, and also may miss hazards/threats resulting from combined security/safety concerns. There are some early attempts at this in the literature. Unfortunately, this is not yet a common approach, simply because the relevant "standards" *ISO 26262* and *SAE J3061* deal with the two aspects separately in order to limit their scope during their initial development. Our own work currently is based on compliance with *ISO 26262* and *SAE J3061*. However, we suggest more comprehensive integration in our section on "Future Work" (Section 6.1). In the meantime, we have included a brief look at the literature on integrated hazard and threat analysis in this section.

Systems-Theoretic Process Analysis (STPA), developed by Nancy Leveson, is a well-regarded hazard analysis technique that is focused strictly on ensuring safety [16]. *STPA-Sec* [32] developed by Leveson and Young, is a derivative of STPA in the security domain. STPA-Sec is an extension of STPA considering security aspects in a top-down fashion. However, in striving to integrate safety and security analysis, separate analysis of safety and security does not seem to adequately cover the integrated effects of safety and security. Another method, *STPA-SafeSec* [8] based on STPA, proposes a more unified analysis technique for safety and security. To support the unified approach, STPA-SafeSec defines the component layer diagram and extends the causal factors of security domains. This method considers the cyberattacks on integrity and availability at the component layer. The authors do not show the relationship between safety and security, and how conflicts can be resolved is not explicitly defined.

In [23], the authors developed a method called *SAFE (Systematic Analysis of Faults and Errors)*. In order to combine safety and security, SAFE considers a semantic framework of error "effect" that integrates an adversary model used in security analysis with fault/error categorization used in hazard analysis. This method is a heavily modified form of STPA. Safety and security analysis are also combined in [22]; namely, STPA and NIST SP800-30 [2] are considered to derive the safety constraints and security constraints respectively. The authors use an automatic scheme to detect conflicts and reinforcement. However, they do not define the automatic scheme precisely which is the key mechanism in detecting conflicts. In [17], the *SAHARA (Security Aware Hazard Analysis and Risk Assessment)* method derives a measure of the security impact on the *"Automotive Safety Integrity Levels" (ASILs)*. This approach uses STRIDE (**S**poofing identity, **T**ampering with data, **R**epudiation, **I**nformation disclosure, **D**enial of service, **E**levation of privilege) to derive *"Security Levels"* to combine with the ASILs based on *ISO 26262*'s HARA (Hazard analysis and risk assessment). Amorim et al [1] use patterns to interlink safety and security in the development process. Some of the authors of that paper were also involved in creating SAHARA, described above.

4 An ACT for Safety & Security of OTA Updates

The primary aim of this work was to develop an ACT that can be used in general to assure safety and security for automotive vehicles, and especially to deal adequately with OTA Updates. We divided the task into two:

1. Step 1 – Develop an ACT for safety and security of automotive vehicles, compliant with both *ISO 26262* and *SAE J3061*;
2. Step 2 – Specialize the previously developed ACT, to include assurance when maintenance is performed using OTA Updates.

This approach was used since both of the relevant standards do not include specific guidance for OTA Updates, and we believe that there is some general guidance we can provide that covers both maintenance implemented at a dealership, or through OTA Updates.

4.1 Assurance of Integrated Safety and Security

Generally, security and safety are considered separate disciplines because of their own regulations, standards and methodologies [6]. A concept is gaining momentum that security and safety are closely interconnected. Nowadays it is not acceptable to assume that a cyber-physical system is immune to threats and it is not feasible to assure the safety of the cyber-physical system independent of security. In this regard, a safety case is incomplete and unconvincing without consideration of the impact of security. In [6], the authors emphasize that the impact of security on the safety case should be explicitly mentioned to make the system safe and secure. In [21], the authors describe a layered assurance approach that combines safety and security.

4.2 Step 1 – An Automotive ACT for Safety & Security

Figure 2 shows the top-level of a security informed safety ACT for "$< X >$ *considered as an ISO 26262 item/SAE J3061 feature, delivers the behaviour required and does not adversely affect the safety of the vehicle, nor does it create security vulnerabilities in the vehicle, over its expected lifetime in its intended environment*". (We have not included "context", "assumptions" and the content of "strategy" nodes in the diagrams, in the interest of saving space.) Six sub-claims support the top level claim. All six sub-claims deal with safety and security issues together with consistent interaction. The tabs on the top left of a claim node indicate that this is a *module*, and the remainder of that argument path can be seen by "opening" that module (in the tool we use, achieved by double clicking the tab). The relevant ISO and SAE clauses/sections are indicated inside a smaller text box within the claim. In terms of software engineering, four argument paths could be shown to adequately support a top claim of safety of a specific system. An informal description of the four top level claims supported by these arguments, would be:

1. The system's requirements are "correct". [**GS** in Figure 2.]
2. The system is implemented to meet its requirements. [**GR** in Figure 2.]
3. The system is safe even when maintenance is performed. [**GPM** in Figure 2.]
4. The system is operated within its operational assumptions. [**GA** in Figure 2.]

ISO 26262 and *SAE J3061* take a similar approach, and add two more claims:

5. Compliance with configuration management requirements. [**GC** in Figure 2.]
6. Compliance with change management requirements. [**GCM** in Figure 2.]

Roughly, the argument that the conjunction of 1,2,3,4 implies safe and secure, follows from the fact that we can think of these claims as:

1. validation,
2. verification,
3. safe/robust with respect to change,
4. operated within known bounds, respectively.

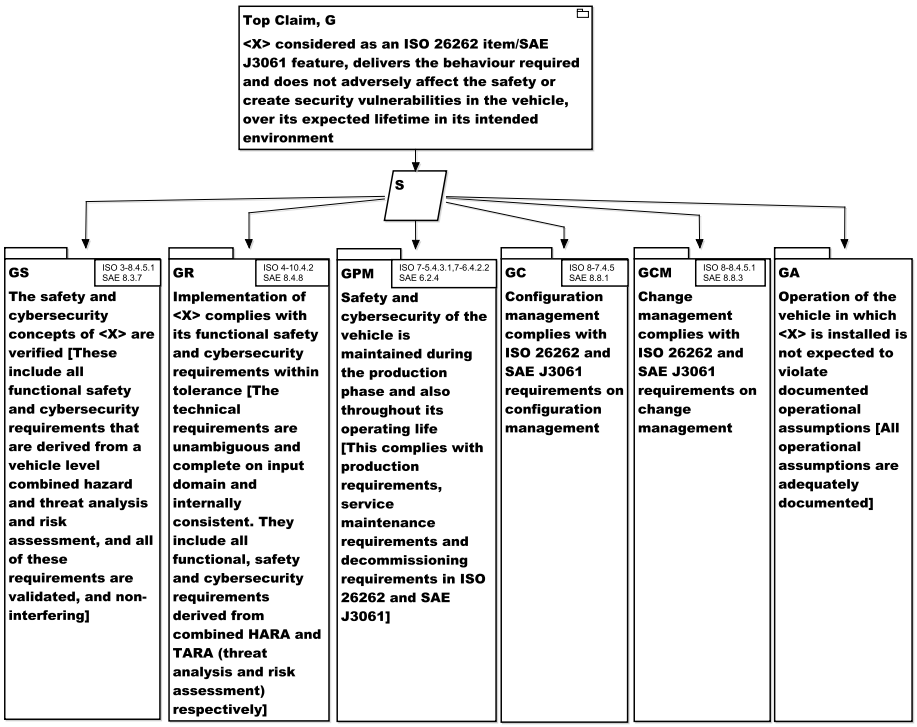

Fig. 2. Top-Level of a Safety and Security ACT

Of particular interest in this template is the argument path leading off the claim "GPM" in Figure 2. This path shows how we, through compliance with the standards, argue the safety of *maintenance* throughout the life of the vehicle. Part 7 of *ISO 26262* and Section 6 of *SAE J3061* define maintenance requirements on production, and operation. We have highlighted this path because it is of central importance in arguing the safety and security of OTA Updates. Figure 3 shows a slice of "GPM" developed from *ISO 26262* and *SAE J3061*. The structure is largely dictated by the structure of the standards. For example, the safety argument is contained in the GPM1 branch, and the security argument in the GPM2 branch. *ISO 26262* describes requirements on production, maintenance, and decommissioning. One option would have been to split these at the sub-claim level shown in Figure 2. We chose to combine them in a single claim, and so the premises for GPM1 are GPM1.1 (production), GPM1.2 (maintenance) and GPM1.3 (decommissioning). Similarly, the premises for GPM2 are GPM2.1 (production), GPM2.2 (maintenance) and GPM2.3 (decommissioning). The decommissioning claim (in module-GPM2.3) was further decomposed into several sub-claims to assure the required cybersecurity properties during decommissioning. They are not shown in the diagram as they are not relevant within

the context of this paper. Figure 3 shows compliance with the ISO and SAE standards **before** any specialization for OTA Updates. It is reasonably obvious that OTA Updates will affect claim GPM1.2 and its argument path (safety) and claim GPM2.2 and its argument path (cyber-security) – primarily the GPM2.2.1 argument path.

4.3 Step 2 – An Automotive ACT for Safety & Security that Includes OTA Updates

In order to include OTA Updates explicitly in the ACT, we have to analyze exactly what is different between traditional at the dealership maintenance, and OTA Update maintenance. This involves both hazard and threat analyses. OTA Updates introduce both safety and security vulnerabilities. In terms of safety, OTA Updates are performed remotely, without the aid of a knowledgeable technician who would be responsible for testing the update. Clearly, the update will have been thoroughly tested by the manufacturer, but there are significant issues of *completeness* that complicate this task. An obvious example is the malfunctioning heads-up display discussed in the Introduction. In terms of security, *SAE J3061* describes in general how to protect the vehicle from cyber-security attacks. The guidebook does not explicitly consider what is necessary when maintenance is performed OTA. We want to include the option of OTA Updates in our ACT. To do this, we used the work reported in the design of Uptane [13,15] as the basis of the OTA-specific arguments in the ACT, as far as security is concerned. Once we have a design in mind (and Uptane is sufficiently generic in terms of identification of communication channels), we are in a position to generate threats and mitigations that can be used as a base for the assurance case argument.

When analyzing OTA Updates, not only must the security of data be considered, but the protocols that handle this data must also be considered. We note that relevant attacks consistently target and exploit weaknesses of four main security properties: *confidentiality*, *integrity*, *availability*, and *authenticity* [26]. By adequately protecting these four main properties, which have been at the root of all known attacks, it is possible to provide security assurance for the system.

The first three of these properties are widely considered to be the most crucial components of information security [27], and are known as the *CIA triad*, and CIA is currently being used to analyze the security requirements of more than one hundred use cases of the connected vehicle proposed by the ARC-IT project funded by the U.S. Department of Transportation [28].

Confidentiality: Confidentiality of communicated information is put at risk by read attacks. Data encryption is suggested to mitigate these attacks [26].

Integrity: The integrity of data can be protected through the use of hashing, cyclic redundancy checks (CRC) and signatures, preferably used in combination.

Availability: A comprehensive backup strategy, anomaly detection, and timeouts are recommended to mitigate these attacks [26].

Although the CIA triad are considered the most crucial components of information security, they are not enough to completely secure the system. The

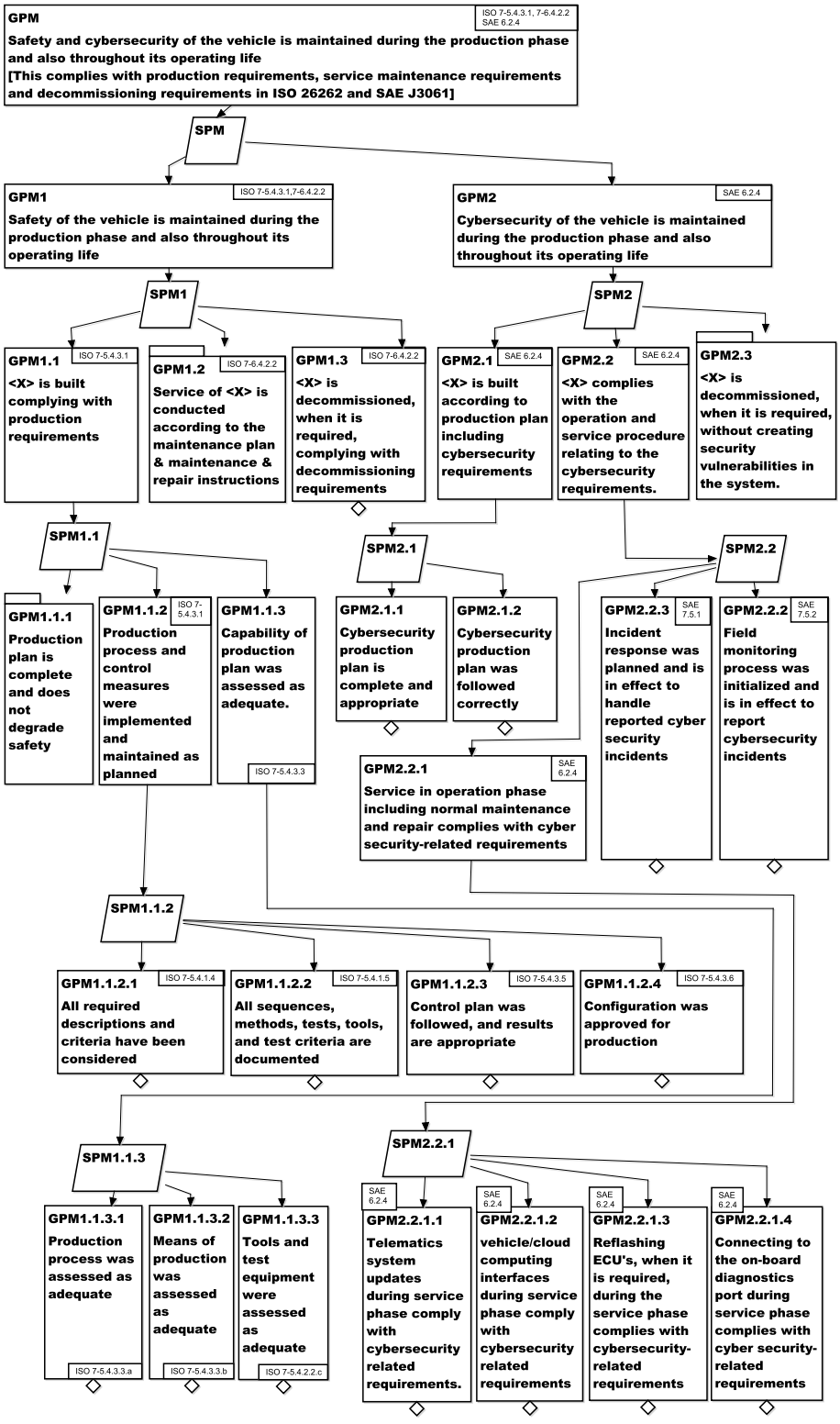

Fig. 3. Extract of Assurance Case for Maintenance of Automotive Vehicles (GPM)

STRIDE Threat Model from Microsoft [19] recommends protection of Authenticity, Authorization, and Non-repudiation.
Authenticity: Authenticity ensures that the data received comes from a trustworthy source. This protects against man in the middle (MITM) and spoofing attacks [26].
Authorization: Authorization prevents unprivileged parties gaining access [31].
Non-repudiation: Maintaining secure logs of activities and the entities to which they are attributed protects non-repudiation scenarios [31].

We also need to consider two generic security measures – **private key protection** and **version control**. Private key protection can help prevent "key extraction" [26], and version control is essential in general, but can also help protect against installation of an older version of software.

There exist a number of tools and methodologies for classifying and managing security related threats. Many of these are outlined in *SAE J3061*. We chose to use Microsoft's threat modelling tool which performs threat analysis using STRIDE [19] and a data flow diagram of the system [18]. STRIDE classifies attacks (threats) into six categories – *Spoofing identity*, *Tampering with data*, *Repudiation*, *Information disclosure*, *Denial of service*, and *Elevation of privilege*.

For each type of threat presented by STRIDE, Microsoft suggests a security property countermeasure.

The NCC group [20] created a customized template for the automotive domain to perform threat analysis using STRIDE, and we created a data flow diagram that describes the communication flow as input to STRIDE. We modelled our data flow diagram on an Uptane design (Figure 1 in [13]). Our data flow diagram of a partial vehicle network illustrating OTA Updates is shown in Figure 4. We used the NCC template together with the data flow diagram in Figure 4 to analyze OTA Updates for the connected car, to generate threats and corresponding mitigations to include in our ACT.

A slice of GPM specialized for OTA Updates using the results from the STRIDE analysis, is shown in Figure 5.

5 Example Usage of the ACT for OTA Updates

We used this slice of the ACT to explore what we would need to do to develop a safe and secure OTA Update design. Thanks to the extensive work in the Uptane project, we could use their design and a python implementation as an example. We found that part of the implementation did not satisfy one of the threat mitigation requirements in the acceptance criteria of the ACT (see Figure 5).

In particular, Threat 2 in EPM 2.2.1.1.1.a.2.1 refers to a MITM threat. This may lead to a vulnerability in the Uptane implementation. The suggested mitigation strategy is that communication must be secured (using TLS or cryptographically signed). In the sample implementation, requests from the primary ECU to the OEM's time server for an updated timestamp are sent as unsigned plain text. (The OEM time server is included in the OEM Update Server in Figure 4). Although the communication is just a pseudorandom nonce from each

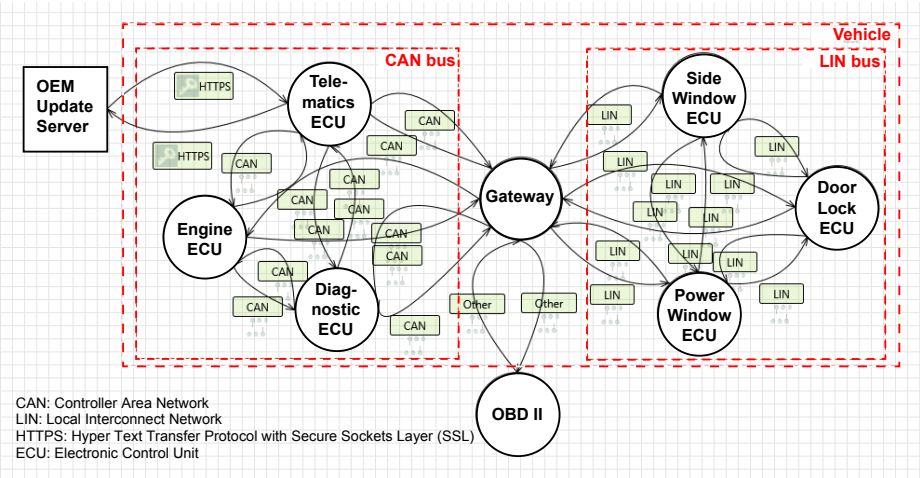

Fig. 4. Data Flow Diagram of a partial vehicle network based on Uptane [13]

secondary ECU, this allows MITM agents to alter the communication as they see fit, and force the system into an unexpected state. Depending on a vendor's implementation, attacks such as a buffer overflow could be possible. In this case, editing the packet to contain no nonces, then allowing it to go through, causes the primary ECU to ignore the updated time. However, it will then make its next request to the time server without sending any nonces, at which point the MITM can inject a subset of the previously blocked nonces, and the primary ECU will accept the reply from the time server. The primary ECU will then pass the message from the time server along to all the secondary ECUs, but since the MITM manipulated the exchange to only contain a subset of nonces, only secondary ECUs in this selected sub set will accept the updated time. If a vendor decides to implement a check for a recent timestamp from the time server on each secondary ECU before installing an update, a Mixed Bundle Attack could be possible. The ACT suggests mitigating this vulnerability by signing the packet of nonces from the primary ECU to the time server. If the developer does this, this specific MITM attack can be mitigated.

6 Conclusion

We have demonstrated that ACTs can be designed to integrate functional safety and security for automotive vehicles. As a basis for such a template, we start with claims and evidence derived from *ISO 26262* and *SAE J3061* using principles we developed for just such a process. We then integrate into that ACT, claims and evidence based on our knowledge and derived from additional analyses. In particular, we specialized the ACT to include specific arguments that apply when maintenance is performed using OTA Updates. To illustrate the value of using these templates in automotive development, we used the OTA specific

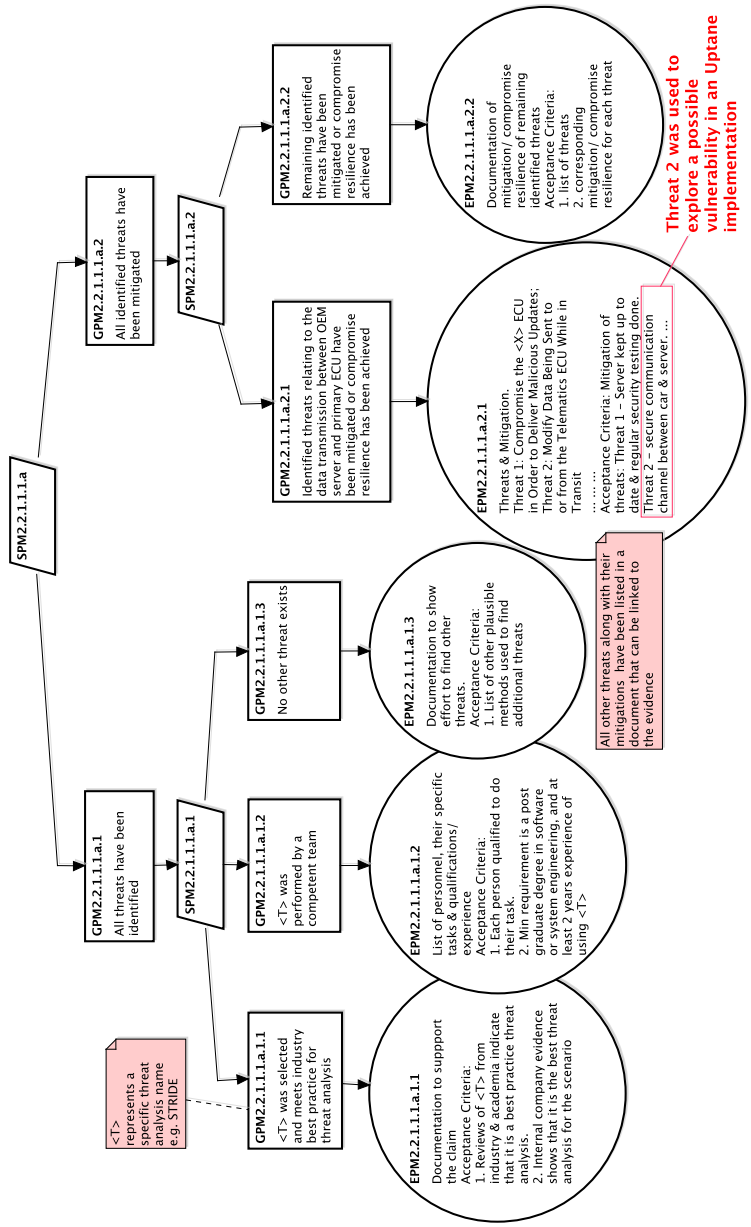

Fig. 5. Slice of OTA Update ACT

template to show that it may help developers mitigate security threats, during development of the vehicle.

6.1 Future Work

We are exploring how to better integrate functional safety and security. As mentioned earlier, one of the best ways is to integrate the hazard and threat analyses. If we achieve this, we will then re-evaluate the argument in the assurance case. This should eventually result in changes to *ISO 26262* and *SAE J3061* – or even better, the integration of cyber-security into *ISO 26262*. We intend to derive more examples of evidence, and especially acceptance criteria for that evidence. In this paper we only considered the integration of security of OTA and functional safety in a combined ACT. Currently ISO is developing a new standard, ISO 21448, on "Road vehicles – Safety of the intended functionality" [12]. Rather than the safety of the system in the presence of failures, this standard concerns itself with the safety of systems when they are functioning correctly. Once the standard becomes available we intend to develop an explicit ACT for it applying the same method used for ISO 26262 [7]. We could then integrate the SOTIF ACT with the ACT proposed in this paper. Finally, although we have made quite a lot of progress in working out *how* to develop these ACTs, we believe there is still more work to be done in this regard, and we are starting to look at syntax and semantics for ACTs in order to build effective tools.

References

1. Amorim, T., Martin, H., Ma, Z., Schmittner, C., Schneider, D., Macher, G., Winkler, B., Krammer, M., Kreiner, C.: Systematic pattern approach for safety and security co-engineering in the automotive domain. In: SAFECOMP. pp. 329–342. Springer (2017)
2. Aroms, E., et al.: NIST Special Publication 800-30 Risk Management Guide for Information Technology Systems (2012)
3. Barber, A.: Status of work in process on ISO/SAE 21434 Automotive Cybersecurity Standard. `https://www.sans.org/summit-archives/file/summit-archive-1525889601.pdf`, accessed: 2018-05-28
4. BBC News: Faulty update breaks lexus cars' maps and radio systems. `http://www.bbc.com/news/technology-36478641`, accessed: 2016-06-08
5. Bloomfield, R., Bishop, P., Jones, C., Froome, P.: ASCAD, Adelard Safety Case Development Manual. Adelard, 1998. ISBN 0-9533771-0 5 (1998)
6. Bloomfield, R., Netkachova, K., Stroud, R.: Security-informed safety: if it's not secure, it's not safe. In: Int. Workshop on Software Engineering for Resilient Systems. pp. 17–32. Springer (2013)
7. Chowdhury, T., Lin, C.W., Kim, B., Lawford, M., Shiraishi, S., Wassyng, A.: Principles for Systematic Development of an Assurance Case Template from ISO 26262. In: IEEE Int. Symp. on Software Reliability Engineering. pp. 69–72 (Oct 2017)
8. Friedberg, I., McLaughlin, K., Smith, P., Laverty, D., Sezer, S.: STPA-SafeSec: safety and security analysis for cyber-physical systems. Journal of Information Security and Applications 34, 183–196 (2017)
9. Graydon, P., Knight, J., Strunk, E.: Assurance based development of critical systems. In: Dependable Systems and Networks, 2007. DSN '07. 37th Annual IEEE/IFIP International Conference on. pp. 347–357 (June 2007)

10. ISO: 26262: Road vehicles-Functional safety. International Standard ISO 26262 (2011)
11. ISO/SAE AWI: 21434: Road vehicles-Cybersecurity Engineering, [Under development]
12. ISO/WD PAS: 21448: Road vehicles – Safety of the intended functionality, [Under development]
13. Karthik, T., Brown, A., Awwad, S., McCoy, D., Bielawski, R., Mott, C., Lauzon, S., Weimerskirch, A., Cappos, J.: Uptane: Securing software updates for automobiles. Int. Conf. Embeded Security in Car. pp. 1–11 (2016)
14. Kelly, T.: Arguing Safety – A Systematic Approach to Managing Safety Cases. Ph.D. thesis, University of York (September 1998)
15. Lauzon, S.: Secure software updates for automotive systems: Introduction to the Uptane SOTA solution (May 2017)
16. Leveson, N.: Engineering a safer world: Systems thinking applied to safety. MIT press (2011)
17. Macher, G., Armengaud, E., Brenner, E., Kreiner, C.: Threat and risk assessment methodologies in the automotive domain. Procedia computer science 83, 1288–1294 (2016)
18. Microsoft: Microsoft threat modeling tool. `https://www.microsoft.com/en-us/download/details.aspx?id=49168`, accessed: 2017-09-20
19. Microsoft: The STRIDE threat model. `https://msdn.microsoft.com/en-us/library/ee823878(v=cs.20).aspx`, accessed: 2017-09-20
20. NCC Group: The Automotive Threat Modeling Template. `https://www.nccgroup.trust/us/about-us/newsroom-and-events/blog/2016/july/the-automotive-threat-modeling-template/`, accessed: 2017-09-20
21. Netkachova, K., Müller, K., Paulitsch, M., Bloomfield, R.: Security-informed safety case approach to analysing MILS systems (2015)
22. Pereira, D., Hirata, C., Pagliares, R., Nadjm-Tehrani, S.: Towards combined safety and security constraints analysis. In: SAFECOMP. pp. 70–80. Springer (2017)
23. Procter, S., Vasserman, E.Y., Hatcliff, J.: Safe and secure: Deeply integrating security in a new hazard analysis. In: ARES. p. 66. ACM (2017)
24. SAE International: SAE J3061-Cybersecurity Guidebook for Cyber-Physical Automotive Systems. SAE-Society of Automotive Engineers (2016)
25. Shostack, A.: Threat modeling: Designing for security. John Wiley & Sons (2014)
26. Spaan, R., Batina, L., Schwabe, P., Verheijden, S.: Secure updates in automotive systems. Nijmegen: Radboud University pp. 1–71 (2016)
27. Summers, A., Tickner, C.: What is Security Analysis. `http://www.doc.ic.ac.uk/~ajs300/security/CIA.htm`, accessed: 2017-09-20
28. US Dept of Transportation: Architecture reference for cooperative and intelligent transportation. `https://local.iteris.com/arc-it/`, accessed: 2018-02-24
29. Wassyng, A., Joannou, P., Lawford, M., Maibaum, T.S., Singh, N.K.: Chapter 13 New Standards for Trustworthy Cyber-Physical Systems. In: Trustworthy Cyber-Physical Systems Engineering, pp. 337–368. CRC Press (2016)
30. Wassyng, A., Maibaum, T., Lawford, M., Bherer, H.: Software certification: Is there a case against safety cases? In: Calinescu, R., Jackson, E. (eds.) Foundations of Computer Software. Modeling, Development, and Verification of Adaptive Systems. pp. 206–227. Springer, Berlin, Heidelberg (2011)
31. Webtrend: Threat modeling with STRIDE. `https://www.webtrends.com/blog/2015/04/threat-modeling-with-stride/`, accessed: 2017-09-20
32. Young, W., Leveson, N.: Systems thinking for safety and security. In: Proc. of the 29th Annual Computer Security Applications Conference. pp. 1–8. ACM (2013)