Rethinking transport
27–30 April 2020

Proceedings of 8th Transport Research Arena TRA 2020, April 27-30, 2020, Helsinki, Finland

Automated driving systems meet edge computing:

A security perspective

Tahsin C. M. Dönmez, Yuan Zhang, Ethiopia Nigussie,

Antti Hakkala, and Jouni Isoaho

Department of Future Technologies, University of Turku, 20014 Turun yliopisto, Turku, Finland

Abstract

This work presents an overview of security challenges associated with Automated Driving Systems (ADSs), covering on- and off-board computations, trustworthiness of data sources, and communications. We focus on the security of computations and emergent involvement of the edge computing paradigm in ADSs. Analysis of ADS and edge computing security reveals that ensuring the integrity of offloaded computations and availability of the offloading service are major security challenges that must be addressed if edge computing is to play a significant role in ADSs. Verifiable computation is investigated as a security solution to ensure integrity of offloaded ADS computations, and a method for deciding on the adequateness of a verifiable computation scheme for ADS applications is proposed.

Keywords: Automated Driving System; Edge Computing; Security; Verifiable Computation; Self-driving; Driverless

1. Introduction

Automated driving has an ambitious goal: to replace human operators of vehicles with intelligent (non-human) machines that possess better and more diverse sensory skills, better decision skills, and better driving skills (e.g. faster and more precise steering, faster reaction times). Automated Driving System (ADS) is the term adopted by Society of Automotive Engineers (SAE), in favor of alternatives such as autonomous, unmanned, self-driving, driverless, and robot vehicles. There are also several frequently used terms that are conceptually very relevant, such as connected vehicles, and Intelligent Transport Systems (ITS) defined by the European Union directive 2010/40/EU. Theoretically a vehicle capable of automated driving does not have to be connected; however, from a practical point of view connectedness seems to be mandatory for achieving higher degrees of automation, especially when the limitations of today's on-board sensors and computers are considered. A connected vehicle possesses vehicle-to-everything (V2X) communication capabilities. V2X communications include, but are not limited to, vehicle-to-infrastructure (V2I) and vehicle-to-vehicle (V2V) communications.

The term ADS is quite ambiguous when the degree of automation is not specified. One attempt at defining the different levels of automation is SAE's taxonomy (SAE International, 2016) which define six levels of driving automation for motor vehicles on roadways: 0 - No Automation, 1 - Driver Assistance, 2 - Partial Automation, 3 - Conditional Automation, 4 - High Automation, 5 - Full Automation.

Traffic safety is a central driver for autonomous vehicles, as computers are at least on paper better drivers than people and can be trained on data from other sources. For example, Waymo has tested their autonomous vehicle platform for over 5 million miles on real roads, and have simulated billions more (Waymo, 2018). Human operators could be freed from driving tasks, and people could spend less time in traffic due to increased traffic flow. Vehicles could last longer, and fuel consumption could be reduced due to better driving practices. Mobility for children and disabled people could be enhanced. Additionally, automation of driving will likely lead to shared vehicles, reducing the total number of vehicles and parking space needed, and all the associated environmental costs. Such examples of societal changes may be reality in the future (Heimo et al., 2019), but the development towards service-based models instead of car ownership may in turn create new security and safety issues for autonomous vehicles (Hakkala & Heimo, 2019).

Several technical challenges are yet to be solved for fully automated driving. Edge computing paradigm presents itself as an enabler for tackling some of these challenges. In this work, we use the term edge computing to refer to the general idea of making computational, storage, and network resources available to their users at locations closer to them than cloud data centers. Frequently used names that are closely related to this general idea are Multi-Access Edge Computing (MEC)[1] coined by the European Telecommunications Standards Institute (ETSI), Fog Computing coined by Cisco, and Cloudlets. Although edge computing can exist on its own, historically it emerged as a supplement to cloud computing. As resources are moved closer to users, an important trade-off presents itself: the amount of resources that can be made available to a user becomes smaller, but much lower latency and better reliability can be obtained compared to cloud computing. When latency and reliability are sufficiently improved to satisfy the strict requirements of ADSs, the opportunity to offload mission-critical ADS computations to edge computing devices (ECDs) arises.[2]

As the degree of connectedness increases for the vehicles, there is growing concern about the possibility of remote attacks targeting vehicles. On the other hand, as the degree of automation increases, and control is taken away from humans, trust issues surface. Resolution of security and trust issues will play an important role in the development and acceptance of automated driving systems. Involvement of edge computing in ADSs promises improvements and new capabilities, but also creates new security challenges which must be addressed. While security aspects of ADSs have attracted much attention in the research community, security problems which may arise due to the involvement of edge computing has not yet received the same level of attention.

In this paper we contribute to the filling of this gap by highlighting these relatively ignored security challenges by consolidating the security challenges associated with ADS and further identifying main security challenges arising

[1] Before a renaming in 2017, the name Mobile Edge Computing (MEC) was used.

[2] Due to the relative scarcity of resources on the edge, it often makes sense to have a multi-tier architecture where either edge or cloud resources can be utilized depending on the circumstances. This is also true for ADSs.

from the involvement of edge computing, specifically due to offloading of ADS computations. In addition, we propose a method for deciding on the adequateness of a verifiable computation scheme for ADS applications.

This paper is organized as follows. Section 2 presents related work. Section 3 presents an overview of ADS and edge computing security. Section 4 discusses the motivation for involving edge computing in ADSs, as well as the associated challenges. Section 5 discusses verifiable offloading of ADS computations, and adequateness of a verifiable computation scheme for offloading mission-critical ADS computations. Section 6 concludes the paper.

2. Related Work

There are many papers which discuss the possibility of offloading ADS computations at a conceptual level. However, the number of works which go deeper are not too many. Some of these are mentioned below. Among these, a few of them acknowledge the security and privacy challenges, but their discussions of security are very brief, as security is not the main focus in any of these works.

Zhang et al. (2018) propose an edge computing-based platform OpenVDAP which consists of an OBC, a vehicle-dedicated operating system, and an edge-aware application library. OpenVDAP enables offloading of ADS computations such as object detection to edge and cloud computing devices.

Sasaki et al. (2016) propose an "infrastructure-based vehicle control system" where driving decisions are offloaded to an edge or cloud computing device. They also present a comparison of driving performance for edge and cloud computing cases.

Qiao et al. (2018) propose a collaborative task offloading scheme which allows offloading to other vehicles as well as ECDs which are part of the infrastructure. Their analysis suggests that perception reaction times are less when various offloading schemes are deployed, compared to the case where three-dimensional scene reconstruction tasks are computed on-board.

Tang et al. (2018) report on an edge computing framework for automated vehicles. Their prototype supports real-time localization through Simultaneous Localization And Mapping (SLAM), real-time obstacle detection through computer vision, and speech recognition for user interaction. Computer vision and speech recognition tasks can be dynamically offloaded to an ECD through the edge-cloud coordination unit in their π-Edge Architecture.

Verifiable computation is a notion introduced and formalized by Gennaro, Gentry, and Parno (2010). Shan, Ren, Blanton, & Wang (2018) present a survey of verifiable computation schemes with focus on solutions targeting specific solutions. Demirel, Schabhser, & Buchmann (2017) categorize general purpose solutions under four major categories: (1) proof and argument-based solutions; (2) solutions based on fully homomorphic encryption; (3) homomorphic authenticators; (4) solutions based on functional encryption and functional signatures. Each major category is further divided into subcategories based on the underlying mathematical constructions. Yu, Yan, & Vasilakos (2017) also focus on general-purpose solutions. They identify a set of desirable properties (non-essential requirements) of verifiable computation and analyze recent proposed solutions in terms of these.

3. Security Challenges

3.1. Automated Driving Systems

The very purpose of vehicles is fast transportation of passengers and objects, and wrong decisions concerning driving tend to result in violent collisions which are often fatal for living things. Consequently, risks associated with autonomous driving can be very high-impact and life-threatening. It is crucial to have built-in security for any system involved in mission-critical decisions concerning automated driving. The security of ADSs is currently under heavy research, as is shown by recent surveys into the field (Cui, Liew, Sabaliauskaite, & Zhou, 2018; Le, den Hartog, & Zannone, 2018; Parkinson, Ward, Wilson, & Miller, 2017; Torre, Rad, & Choo, 2018). In this paper we categorize security challenges associated with ADSs into three groups:

(1) **Challenges related to processed data**: Data processed for making driving decisions may include data generated by vehicle's on-board sensors, as well as data generated by infrastructure (e.g. roadway

sensors), road users (e.g. vehicles, pedestrians), and external entities (e.g. GNSS). Ensuring correctness, availability, and confidentiality of generated data is a difficult task even when all communications are assumed to be perfectly secured. Availability of sensors may be compromised by physical attacks, correctness of generated data may be compromised by tampering with the sensors or the sensed medium, and there may be reasons not to reveal some input data to the processing party, for example when one vehicle processes data generated by another vehicle. Trusting of the various data generators is a necessity, but justification of that trust turns out to be a hard problem. Possible security solutions to address these challenges are physical protection, use of secure cryptoprocessors with sensing hardware, privacy-preserving computations, and networks of trust. Necessity for data protection is not limited to raw sensor readings. Due to the continuous digitalization of transportation systems, other contextual data, including vehicle users' data, become increasingly involved. User data may be processed and stored by various systems, including ADSs. Consequently, regulations related to data protection and anonymization, such as The General Data Protection Regulation (EU) 2016/679, are as relevant to ADSs as the various automotive safety regulations. An ADS must handle the storage and ownership issues about such data, in accordance with the regulations that apply. Finally, intellectual property rights (IPR) protection can become an issue with open data, as it is possible to infer system functionality and internal algorithms from data processed by the system.

(2) **Challenges related to communication of data**: Whenever a network protocol is integrated into a system, overall system complexity increases, and the network protocol brings with it its own security vulnerabilities, in addition to its security features. For an ADS, there are competing alternatives for both in-vehicle and V2X communications. In-vehicle communication between various sensors, actuators, electronic control units (ECU), and on-board computer(s) (OBC) may involve one or more of Controller Area Network (CAN), FlexRay, Byteflight, Automotive Ethernet, Local Interconnect Network (LIN), Low-Voltage Differential Signaling (LVDS), and Media Oriented Systems Transport (MOST) as communication technologies. There are two competing alternatives for V2X communications. First alternative is referred to as Intelligent Transport Systems operating in the 5 GHz frequency range (ITS-G5) in Europe, and as Dedicated Short Range Communications (DSRC) in the United States. Standards associated with this alternative are the IEEE Wireless Access in Vehicular Environments (WAVE) standard, and physical layer and medium access control specifications 802.11p and 802.11bd. Second alternative is Cellular V2X (C-V2X). Current C-V2X version is LTE-V2X which is part of 3GPP releases 14 and 15. NR-V2X is the newer version planned for future releases, and it will work with 5G New Radio (5G NR). C-V2X supports two communication modes. It supports direct communication between devices (e.g. V2V, V2I) at sub-kilometer ranges using ITS bands. This communication mode is independent of cellular network. It also supports vehicle-to-network (V2N) communication at longer ranges. Resilience of the physical layer against jamming is very important from a security point of view and should be taken into consideration when evaluating the competing V2X technologies. In case of V2X communications, short link duration due to high mobility and the ad-hoc nature of the network make association and authentication more challenging. In case of in-vehicle communications, most of the above-mentioned protocols predate connected vehicles, which means their designs did not have to address remote attacks as a major threat. In any case, it is important that the protocols chosen for integration into an ADS have sufficient security features, as well as being secure themselves. These challenges may be addressed by carrying out formal security analysis of the integrated communication protocols, and by deploying intrusion detection and prevention systems (IDPS).

(3) **Challenges related to processing of data**: Among the three groups of security challenges associated with ADSs, those related to the security of the computations themselves seem to have attracted the least attention in the research community. Mission-critical ADS computations may be performed on-board the vehicle, or they may be offloaded [3] to an edge or cloud computing device. Different security considerations become relevant depending on the location and effect of the computation. Therefore, we think it is useful to start by categorizing ADS computations. An ADS may involve computations from all three categories. The categorization used in this work is given in Table 1. Figure 1 presents an overview of the computations involved in an ADS. For computations in the On-board category, the main challenge is keeping the OBC available at all times and ensuring the integrity of computations on it. It is important to include security engineering throughout the development lifecycle of an OBC. While it may be

[3] The words offloading, delegation and outsourcing are used interchangeably in the literature.

understandable for older designs to ignore connectedness, today there is no excuse for designing an OBC which do not have built-in security mechanisms to address network-based attacks. It is particularly important to have a strong access control mechanism, so that malicious access to the OBC via a non-critical system does not compromise mission-critical computations. Depending on the complexity of the OBC, availability of customized security software such as anti-virus and capability for automatic security updates may be desirable.

Table 1: Categories for ADS computations based on the location and effect of computations.

Category	Location	Result used by
On-board	OBC	single vehicle
Offloaded	ECD	single vehicle
Shared	ECD	multiple vehicles

Likely candidates for computations in the Offloaded and Shared categories are sensor fusion, machine learning, image processing, path planning, trajectory planning, and other planning and decision-making tasks. Shared computations differ from Offloaded computations in that: (1) computation results may be used by multiple vehicles (e.g. all vehicles served by the same ECD); (2) users of the computation result may or may not provide inputs to the computation. If On-board computations correspond to one-intelligence-per-vehicle-on-board, then Offloaded and Shared computations correspond to one-intelligence-per-vehicle-on-the-edge and one-intelligence-for-all-on-the-edge, respectively. ADS computations may be offloaded by other road users and infrastructure as well. A typical example is roadside cameras offloading image processing tasks to ECD.

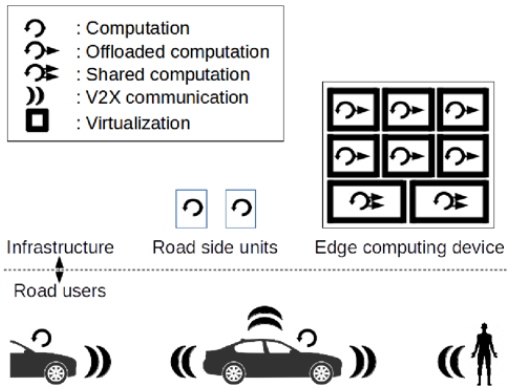

Fig. 1 Various computations involved in the ADS ecosystem

For computations in the Offloaded and Shared categories, security concerns associated with the edge computing paradigm enter the domain of ADSs, where security failures may cost lives. These are discussed in Section 4. Integrity of offloaded mission-critical computations can be ensured with cryptographic guarantees via verifiable computation techniques. These are discussed in Section 5.

An important consideration which concerns all three categories is the transparency of computations. Several machine learning methods (e.g. deep learning) are black-box, meaning that it may not be possible to explain why a certain result was produced. Consequently, for an ADS using a black-box method, there may be situations where neither the ADS itself nor a human expert is able to explain a decision made by the ADS. Considering how costly road accidents can be, there is often great incentive for identifying the responsible party and/or system part. Lack of transparency in ADS computations diminishes system's accountability and has the potential to create liability and legal issues.

3.2. Edge Computing

Cloud and edge computing share many security challenges, and as a result, much of the security research on cloud computing also apply to edge computing. However, the distribution of resources over a wide area makes edge computing security more challenging. For example, a multi-national company may be able to afford building a castle around a data center, but it is not possible to have the same level of physical protection for ECD. The smaller

amount of resources on an ECD compared to a data center does not justify the weaker physical security in all cases. For example, while the amount of data stored on an ECD will be relatively very small, the impact of a data breach due to a compromised ECD may not be. This is even more evident when one considers the case where the ECD is running an ADS application and the integrity of the computations are compromised. In addition to limited physical security, rogue infrastructure is a greater threat in case of edge computing. Both cloud and edge computing security have been studied extensively (Abbas, Zhang, Taherkordi, & Skeie, 2018; Khan, Parkinson, & Qin, 2017; Mukherjee et al., 2017; Roman, Lopez, & Mambo, 2018; Yi, Qin, & Li, 2015; P. Zhang, Zhou, & Fortino, 2018). Table 2 presents a summary of security challenges associated with edge computing in extant literature.

Table 2. Summary of edge computing security challenges.

Challenge	Possible solutions
Physical security of premises	Video surveillance, alarms
Virtualization	VM isolation & monitoring
Variety of actors (e.g. different user types, neighboring ECDs)	Authentication & Access control mechanisms
Uncertainty in behaviors of other actors	Trust management
Network security	Secure communication protocols, IDPSs
Confidentiality of user data	Privacy preserving computations
Correctness of offloaded computations	Verifiable computations, trusted computing

4. Automated Driving Systems Meet Edge Computing

Almost a decade ago, vehicular cloud computing (Abuelela & Olariu, 2010) was proposed as a vision of future where vehicles are both users of cloud services in ITS scenarios, and also service providers who offer their computational and networking capabilities to others. Now edge computing presents itself as an enabler for automated driving systems, thanks to its ability to accommodate applications with strict latency and reliability requirements. Edge Computing concept was adopted as part of the 5G architecture, and the 5G Automotive Association (5GAA) considers it as part of their 5GAA architecture to take part in both low-latency and V2N ADS use cases (5GAA, 2017). The OpenFog Consortium considers ADS as one of the main fog computing use cases (OpenFog Consortium, 2018). Vehicles may play a dual role in edge computation, as computations may be offloaded to vehicles by other vehicles or entities. However, in this work we are mainly interested in the case where vehicles or road infrastructure offload their mission-critical computations to dedicated ECDs.

4.1. Advantages

Possible advantages of offloading ADS computations to ECDs are as follows:

(1) **Less hardware**: Single multi-purpose computing hardware can be shared by many vehicles (and other devices) for various edge computing tasks.
(2) **More powerful hardware**: An ECD can have more powerful hardware compared to a vehicle. This may allow running more demanding algorithms and may lead to better vehicle decisions.
(3) **Easier upgrades and updates**: Upgrading hardware on ECDs is much easier compared to upgrading hardware in each vehicle. Same also applies to software updates. This allows reacting faster to technological advances in hardware and algorithms.
(4) **Ideal location**: Edge is the ideal location for computations which take data from multiple vehicles as input. A typical scenario for this kind of computation is extended sensors, where raw or processed data gathered through local sensors of vehicles (or through sensors of other road users and infrastructure) are exchanged between these entities to address limitations of on-board sensors[4] and enhance perception of the vehicle. Offloading such computations to an ECD would further enhance perception, as the ECD would have a more complete picture of the environment, compared to a vehicle considering shared sensor data only from neighboring vehicles.

[4] Even with sensor fusion, on-board sensors cannot sense behind obstacles.

4.2. Challenges

Below we list the main challenges associated with offloading ADS computations to ECDs. While the first four items are related to non-security concepts, they are relevant to the discussion because they impose severe constraints on any security solution.

(1) **Latency**: For most ADS applications end-to-end latency requirements are in the 5-100 ms range (3GPP, 2018). While propagation delays are negligible for the relatively short (a few kilometers at most) single-hop links, transmission delays pose the real challenge. In case of offloading from vehicle to an ECD, total transmission delay would increase in proportion to the size of inputs and outputs for the computation. The challenge becomes clear when one considers the fact that on-board cameras and LIDARs alone can generate hundreds of megabytes of data per second. Computation times do not pose an additional challenge in case of non-secure offloading, as the same computation is carried out also in the no offloading case. However, in case of secure offloading, running times tend to increase significantly. Moreover, any scheduling delay in the ECD would also contribute to the perception-response delay of the ADS.

(2) **Reliability**: A chain is as strong as its weakest link, and ADS applications can have very high reliability requirements (e.g. 99.999%) (3GPP, 2018). When computations are offloaded from vehicle to ECD, both the link between vehicle and ECD (V2E link), and ECD itself must be able to satisfy the reliability requirements of the ADS application. Reliability is a challenge to offloading, but at the same time offloading can be a solution for overall ADS reliability. Vehicles are likely to be equipped with redundant computing hardware to increase reliability, and an on-demand offloading mechanism can replace the redundant hardware.

(3) **Scalability**: It is important for ADSs to be able to scale well for high vehicle densities. With five lanes in each direction, and three such highways intersecting, one can have more than 4000 vehicles per square kilometer (3GPP, 2018). With the inclusion of ECDs, V2E links add to the connection density of the ADS. Moreover, ECDs must be capable of serving the vehicles in range without introducing significant scheduling delays.

(4) **Mobility**: V2E links do not pose a new challenge in terms of mobility. Supporting high vehicle speeds (e.g. 250 km/h) is a requirement for the V2I links between RSUs and vehicles, with or without the introduction of ECDs. On the other hand, roaming and handover capabilities do enter the picture as a new challenge. For example, a vehicle may offload a computation task to an ECD *ECD*1, and as it moves out of range of *ECD*1 it might be handed over to another ECD *ECD*2. Moreover, the handover may happen before the vehicle gets the chance to receive the computation result. To achieve the degree of mobility required by vehicles, communication of results or transfer of whole virtual machines may be necessary from one ECD to its neighbors, as well as anticipation of handovers. These mechanisms should also respect the reliability and latency requirements of the ADS.

(5) **Security**: Two main security challenges must be addressed before mission-critical ADS computations can be offloaded to ECDs. The first challenge is the prevention of attacks against integrity of computations.[5] The vehicle (or any other entity) which offloads the computation must be ensured about the correctness of the result. Automated driving brings several liability and trust issues with it, even without the involvement of edge computing. Liability issues regarding platform security, safety and reliability in ADSs become complex especially when multiple parties are involved in providing and using the shared hardware, software, or connectivity services. This may even adversely affect the development and adoption of edge computing in ADS domain. Trust is a major factor in achieving user acceptance. A survey by Rödel et al. (2014) suggests that user acceptance of ADSs tends to decrease as level of automation increases. With the involvement of the ECD, passengers and pedestrians will be required to trust the various network operators or other entities that administer the ECDs, in addition to the manufacturer of the vehicle. This could have a negative impact on the user acceptance of ADSs which utilize edge computing. The phenomenon of *forced trust* (Hakkala, Heimo, Hyrynsalmi, & Kimppa, 2018) describes a situation where the end users and/or customers have no other option but to trust all aspects of an ICT system, regardless of how critical a function it performs. Previously this has been discussed in the context of governmental information systems, but autonomous traffic is certainly similar in scope. Security solutions which provide defense mainly against malicious outsiders such as IDPSs would not be enough, especially since there may be incentives for the entity administering ECDs to return

[5] In certain cases, it might be desirable to ensure privacy (of inputs, outputs, or the computation itself) in addition to ensuring the integrity of the offloaded computations.

incorrect results. For example, there might not be enough resources available to perform the computation and returning some value without performing the computation may be preferred over a breach of the Service Level Agreement (SLA), if the service provider thinks it can get away with it. Trusted computing can be used to provide guarantees on the integrity of the execution environment but integrity of the security hardware itself must be assumed. Having cryptographic guarantees on correctness of offloaded mission-critical ADS computations is highly desirable, as using incorrect results may have severe consequences. While verifiable computations can provide the highly desired cryptographic guarantees, the real challenge lies in building a verifiable computation scheme which can satisfy the latency and scalability requirements of ADS applications. The second challenge is the prevention of attacks against the availability of the service provided by the ECD. Such attacks can be directed at the V2E link or at the ECD itself, and relevant security consideration were presented in sections 3.1 and 3.2, respectively.

5. Offloading Verifiable ADS Computations

Verifiable offloading of computations involves a (possibly computationally weak) offloading party (which will be referred to as the client in the rest of this section), one or more workers who are possibly untrusted by the client, and a verifier. Client and verifier may be the same entity. The client sends the inputs for the computation to the worker, who then computes the outputs of the computation along with additional information which enables the verifier to verify the received result. Usefulness of offloading the computation depends on how much less the cost of verification is compared to the cost of performing the computation, $cost_c$. If the cost of verification is greater than or equal to $cost_c$, the verifier would rather perform the computation itself. This is sometimes referred to as the efficiency property. It is also desirable that, the cost of the verifiable computation to the worker is as close as possible to $cost_c$. There are two other properties that must be satisfied by a verifiable computation. Probability of accepting an incorrect output returned by a worker should be negligible (verifiability), and a correct output returned by the worker should be accepted (correctness). Verifiability is sometimes referred to as checkability in the literature. Verifiability comes in two flavours. Public verifiability is the stronger version which allows verification by any third party, as opposed to only by the verifier as in the case of private verifiability. Section 3.1 defined *Shared* offloaded computations. If clients/verifiers provide inputs to a *Shared* computation, private verifiability is sufficient for each client to verify the result, as each client can use its own secret -which enters the computation as an input- for verification of outputs. On the other hand, if clients/verifiers do not provide inputs to a *Shared* computation, but only use the outputs, then the stronger public verifiability property is required.

Verifiable computation offloading problem is a special case of the more general secure computation offloading problem, where claims are made regarding privacy in addition to integrity of the computation. The integrity of computations must always be protected for offloaded mission-critical ADS computations, but privacy may not be required in all cases. Here privacy may refer to the privacy of computation itself, privacy of inputs to the computation, or privacy of computation outputs. This is the reason why the focus of this work is on the more specific and relatively simpler verifiable computation problem. Even without any privacy requirements, the remaining verifiable computation offloading problem is hard to solve, especially while also aiming to satisfy the requirements of ADS applications.

Different approaches exist to address the verifiable offloading problem. Some solutions target specific computations, whereas others are general-purpose solutions which allow arbitrary computations. General-purpose solutions tend to be less efficient compared to computation-specific solutions. There also exist solutions which are based on additional assumptions. For example, solutions based on replication depend on the assumption of uncorrelated failures. The same computation is offloaded to several workers and correctness of output is decided based on whether it is output by the majority. Another example is remote attestation, which requires the use of trusted hardware. As a result, security of this solution depends on the assumption of trusted hardware.

5.1. Adequateness of verifiable computation schemes for ADS

This section identifies a set of efficiency criteria for verifiable computation schemes. These criteria are then mapped to the basic requirements of ADS applications discussed in Section 4.2. By doing so, we aim to provide a method for estimating the appropriateness of a verifiable computation scheme to be used for offloading mission-critical ADS computations.

Table 3 lists the main efficiency criteria for verifiable computation and indicates which of the involved entities has to bear the associated costs. As an example of offloading a mission-critical ADS computation, we consider sensor fusion as the computation, with raw sensor data generated by on-board sensors as inputs and a list of detections as outputs. The vehicle which needs the sensor fusion output is the client and the verifier, and the worker is an ECD which can communicate with the vehicle via a single-hop V2X communication link. We will refer to this example in the discussion below in order to add some concreteness.

Table 3. Efficiency criteria for verifiable computation.
Client and verifier are assumed to be the same entity, and is denoted by C.
W denotes the worker, and L denotes the link between client and worker.
Check marks indicate which entity has to bear the cost.

		Offload			Verifiable Offload		
Criteria	Notation	C	W	L	C	W	L
Number of communication rounds	N_R						✓
Amount of data communicated	D_C			✓			✓
Random Access Memory (RAM) usage	U_{RAM}		✓		✓	✓	
Non-volatile Memory (NVM) usage	U_{NVM}				✓	✓	
Input preparation running time	T_{IP}				✓		
Output verification running time	T_{OV}				✓		
Computation running time	T_C					✓	

Offloading a computation brings additional costs even when the computation is not verifiable. In this case, the costs associated with U_{RAM} and T_c are moved from client to worker. More importantly, communication link has to bear the cost associated with D_C, which did not exist before. In the example above, this would correspond to transmission of raw sensor data, which is generated at a rate of several hundred megabytes per second, via the V2X link.

In case of verifiable computations, the costs associated with D_C, U_{RAM} and T_C can be significantly larger compared to those in non-verifiable offloading case. Amount of communicated data D_C, for example, might increase due to a necessity to transmit a large proof or due to expansion of each input bit to a security key size during the input preparation step. Changes in U_{RAM} and T_C are also scheme dependent. In the example above, costs associated with U_{RAM} and T_C must be sufficiently low so that a single ECD can simultaneously serve many vehicles. While these costs are moved from the client to the worker, the client still must take part in input preparation, and the verifier has to take part in verification. Costs associated with T_{IP} and T_{OV} must be smaller compared to the cost of performing the non-verifiable computation (efficiency property). Satisfying the efficiency property is a challenge. For example, while verifiable computation schemes can be built upon probabilistically checkable proofs and fully homomorphic encryption, these constructions are yet too inefficient to be of practical use, and verification of even small instances could take hundreds to trillions of years using these constructions (Parno, Howell, Gentry, & Raykova, 2013). N_R mostly depends on the interactive property of the verifiable computation scheme, with non-interactive schemes having smaller N_R values. Whether or not the cost associated with U_{NVM} is significant depends on the particular verifiable computation scheme used. When there is a significant cost, it is incurred on one or both of the worker and the verifier. For example, if the verifiable computation scheme involves the preprocessing model, large amounts of preprocessing material may have to be stored in non-volatile memory of the worker. Or if the scheme involves a proof with a large size, it may have to be stored by the verifier before being processed.

We now connect these costs with the latency, reliability, scalability, and mobility requirements of ADS applications (Section 4.2). What is meant by latency here is the perception-response delay of the ADS. The contribution to latency due to communication is affected by D_C and N_R, and can be expressed as follows:

$$DELAY_{COMM} = (N_R \times L_{PD}) + (D_c \div L_{DR}) \quad (1)$$

where L_{PD} and L_{DR} are propagation delay and data-rate of link L, respectively. The contribution to latency due to

computation can be expressed as follows:

$$DELAY_{COMP} = T_{IP} + T_{OV} + T_C \quad (2)$$

Then, latency can be expressed as:

$$LATENCY = DELAY_{COMM} + DELAY_{COMP} + DELAY_{SC} + DELAY_{HO} \quad (3)$$

$DELAY_{SC}$ and $DELAY_{HO}$ are the scheduling and handover delays, respectively. In the context of the previously mentioned example, scheduling delays occur when the number of vehicles and other clients that require service from the ECD exceed the serving capacity of the ECD. Handover delays are due to the mobility of offloading clients. Serving ECD for a vehicle must change when the vehicle leaves its area of coverage. A handover delay may occur, for example, when the serving ECD changes before the vehicle gets the chance to receive outputs of an offloaded computation. Scalability requirements of an ADS become multifaceted when offloading is involved. Density of vehicles and other clients dictate the capacity of both the communication links and the ECD, as insufficient capacity of either of them may lead to intolerable latency values. Reliability can be expressed in terms of failure rate. When computations are offloaded, failure rate of the ADS becomes:

$$FR_{OFFLOAD} = FR_{NO-OFFLOAD} + (FR_{ECD} - FR_{OBC}) + FR_{LINK} \quad (4)$$

where FR_{ECD}, FR_{OBC}, FR_{LINK} are the failure rates for ECD, OBC, V2X link, respectively. $FR_{NO\text{-}OFFLOAD}$ is the failure rate for the ADS when the computation is performed by the OBC, and not offloaded. Eq. 4 indicates that offloading decreases the reliability of the ADS due to the involvement of an additional communication link, when the ECD and OBC are equally reliable.

In the simplest possible case of offloading (without verifiability), costs are simply moved from client to worker. For verifiable computations it can be advantageous to distribute costs in different ways. Costs can be moved temporally (e.g. to a preprocessing phase), moved spatially (e.g. to a newly introduced entity other than client, worker, and verifier), and amortized. With such redistribution, a formerly impractical scheme can be turned into a verifiable computation scheme of practical value if the right tradeoffs can be made, even when the total costs increase compared to the original scheme. The possibility of such redistribution is also the reason why estimating the efficiency of a verifiable computation scheme by considering solely the efficiency of the underlying mathematical constructions can be inaccurate.

The appropriateness of a chosen verifiable computation scheme can be decided by following the arguments given in this section, when the following are known: (1) latency, reliability, scalability, mobility requirements of the ADS application; (2) values for the efficiency criteria from Table 3 for the chosen verifiable computation scheme; (3) capabilities of the ECD and the communication link.

6. Conclusion

In this work we have presented an overview of security challenges that arise when edge computing is applied to Automated Driving Systems (ADSs). The edge computing paradigm offers improvements and new capabilities for automated driving systems, but it also brings new security challenges into the picture. The two most important security challenges to be addressed are ensuring the correctness of offloaded computations and availability of the offloading service. We studied verifiable computation as a means to ensure integrity of offloaded ADS computations and proposed a method for deciding on the adequateness of a verifiable computation scheme for ADS applications. While mechanisms which can guarantee correctness and strengthen availability already exist, the real challenge will likely lie in the customization of existing mechanisms to satisfy latency, reliability, scalability, and mobility requirements associated with ADSs, or in devising new mechanisms which can do so.

References

3GPP. (2018). *Service requirements for enhanced V2X scenarios*.

5GAA. (2017). *Toward fully connected vehicles: Edge computing for advanced automotive communications*.

Abbas, N., Zhang, Y., Taherkordi, A., & Skeie, T. (2018). Mobile Edge Computing: A Survey. *IEEE Internet of Things Journal*, *5*(1), 450–465. https://doi.org/10.1109/JIOT.2017.2750180

Abuelela, M., & Olariu, S. (2010). Taking VANET to the Clouds. *Proceedings of the 8th International Conference on Advances in Mobile Computing and Multimedia*, 6–13. https://doi.org/10.1145/1971519.1971522

Cui, J., Liew, L. S., Sabaliauskaite, G., & Zhou, F. (2018). A review on safety failures, security attacks, and available countermeasures for autonomous vehicles. *Ad Hoc Networks*. https://doi.org/https://doi.org/10.1016/j.adhoc.2018.12.006

Demirel, D., Schabhser, L., & Buchmann, J. (2017). *Privately and Publicly Verifiable Computing Techniques: A Survey* (1st ed.). Springer Publishing Company, Incorporated.

Gennaro, R., Gentry, C., & Parno, B. (2010). Non-interactive Verifiable Computing: Outsourcing Computation to Untrusted Workers. In T. Rabin (Ed.), *Advances in Cryptology -- CRYPTO 2010* (pp. 465–482). Berlin, Heidelberg: Springer Berlin Heidelberg.

Hakkala, A., & Heimo, O. I. (2019). Automobile Automation and Lifecycle - How Digitalisation and Security Issues Affect the Car as a Product and Service? *IntelliSys2019 Proceedings (To Appear)*.

Hakkala, A., Heimo, O. I., Hyrynsalmi, S., & Kimppa, K. K. (2018). Security, privacy'); drop table users; - and forced trust in the information age? *ACM SIGCAS Computers and Society*, *47*(4), 68–80. https://doi.org/10.1145/3243141.3243150

Heimo, O. I., Kimppa, K. K., & Hakkala, A. (2019). Automated Automobiles in Society. *Proceedings of IEEE Smart World 2019 Conference (To Appear)*.

Khan, S., Parkinson, S., & Qin, Y. (2017). Fog computing security: a review of current applications and security solutions. *Journal of Cloud Computing*, *6*(1), 19. https://doi.org/10.1186/s13677-017-0090-3

Le, V. H., den Hartog, J., & Zannone, N. (2018). Security and privacy for innovative automotive applications: A survey. *Computer Communications*, *132*, 17–41. https://doi.org/https://doi.org/10.1016/j.comcom.2018.09.010

Mukherjee, M., Matam, R., Shu, L., Maglaras, L., Ferrag, M. A., Choudhury, N., & Kumar, V. (2017). Security and Privacy in Fog Computing: Challenges. *IEEE Access*, *5*, 19293–19304. https://doi.org/10.1109/ACCESS.2017.2749422

OpenFog Consortium. (2018). *Fog Computing Use Cases*. Retrieved from https://www.openfogconsortium.org/resources/#use-cases

Parkinson, S., Ward, P., Wilson, K., & Miller, J. (2017). Cyber Threats Facing Autonomous and Connected Vehicles: Future Challenges. *IEEE Transactions on Intelligent Transportation Systems*, *18*(11), 2898–2915. https://doi.org/10.1109/TITS.2017.2665968

Parno, B., Howell, J., Gentry, C., & Raykova, M. (2013). Pinocchio: Nearly Practical Verifiable Computation. *2013 IEEE Symposium on Security and Privacy*, 238–252. https://doi.org/10.1109/SP.2013.47

Qiao, G., Leng, S., Zhang, K., & He, Y. (2018). Collaborative Task Offloading in Vehicular Edge Multi-Access Networks. *IEEE Communications Magazine*, *56*(8), 48–54. https://doi.org/10.1109/MCOM.2018.1701130

Rödel, C., Stadler, S., Meschtscherjakov, A., & Tscheligi, M. (2014). Towards Autonomous Cars: The Effect of Autonomy Levels on Acceptance and User Experience. *Proceedings of the 6th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, 11:1--11:8. https://doi.org/10.1145/2667317.2667330

Roman, R., Lopez, J., & Mambo, M. (2018). Mobile edge computing, Fog et al.: A survey and analysis of security threats and challenges. *Future Generation Computer Systems*, *78*, 680–698. https://doi.org/https://doi.org/10.1016/j.future.2016.11.009

SAE International. (2016). *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*.

Sasaki, K., Suzuki, N., Makido, S., & Nakao, A. (2016). Vehicle control system coordinated between cloud and mobile edge computing. *2016 55th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, 1122–1127. https://doi.org/10.1109/SICE.2016.7749210

Shan, Z., Ren, K., Blanton, M., & Wang, C. (2018). Practical Secure Computation Outsourcing: A Survey. *ACM Comput. Surv.*, *51*(2), 31:1--31:40. https://doi.org/10.1145/3158363

Tang, J., Liu, S., Yu, B., & Shi, W. (2018). *PI-Edge: A Low-Power Edge Computing System for Real-Time Autonomous Driving Services*.

Torre, G. D. La, Rad, P., & Choo, K.-K. R. (2018). Driverless vehicle security: Challenges and future research opportunities. *Future Generation Computer Systems*. https://doi.org/https://doi.org/10.1016/j.future.2017.12.041

Waymo. (2018). *Waymo Safety Report*. Retrieved from https://waymo.com/safety/

Yi, S., Qin, Z., & Li, Q. (2015). Security and Privacy Issues of Fog Computing: A Survey. In K. Xu & H. Zhu (Eds.), *Wireless Algorithms, Systems, and Applications* (pp. 685–695). Cham: Springer International Publishing.

Yu, X., Yan, Z., & Vasilakos, A. (2017). A Survey of Verifiable Computation. *Mobile Networks and Applications*, 1–16. https://doi.org/10.1007/s11036-017-0872-3

Zhang, P., Zhou, M., & Fortino, G. (2018). Security and trust issues in Fog computing: A survey. *Future Generation Computer Systems*, *88*, 16–27. https://doi.org/https://doi.org/10.1016/j.future.2018.05.008

Zhang, Q., Wang, Y., Zhang, X., Liu, L., Wu, X., Shi, W., & Zhong, H. (2018). OpenVDAP: An Open Vehicular Data Analytics Platform for CAVs. *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, 1310–1320. https://doi.org/10.1109/ICDCS.2018.00131