# In-Vehicle Trust Assessment Framework*

Nataša Trkulja$^{1[0000-0003-2945-4869]}$, Artur Hermann$^{1[0009-0004-3406-267X]}$, Ana Petrovska$^{2[0000-0001-6280-2461]}$, Alexander Kiening$^{3[0000-0003-2161-8662]}$, Anderson Ramon Ferraz de Lucena$^{3[0000-0001-8957-784X]}$, and Frank Kargl$^{1[0000-0003-3800-8369]}$

[1] Institute of Distributed Systems, Ulm University, Ulm, Germany
{natasa.trkulja,artur.hermann,frank.kargl}@uni-ulm.de
[2] Huawei Technologies, Munich, Germany ana.petrovska@huawei.com
[3] DENSO AUTOMOTIVE Deutschland GmbH, Eching, Germany
{a.ferraz,a.kiening}@eu.denso.com

**Abstract.** Today's vehicles run various safety-critical applications requiring data input from diverse in-vehicle components. Adaptive Cruise Control (ACC), for example, can rely on the data input from components such as lidar, radar, GNSS, and cameras. Malicious manipulation of any of this data compromises the data integrity and can result in safety incidents or accidents on the road. Security mechanisms like intrusion detection can be in place; however, they cannot reliably assess the consequences of attacks on a system level or for arbitrary subsystems. In this paper, we present a Trust Assessment Framework (TAF) that allows an in-vehicle application in a complex System-of-Systems to assess whether it can trust the integrity of its input data. The TAF assesses the trustworthiness of every component in the data flow chain based on collected evidence. We explain this concept with the example of ACC and showcase two possible implementations of the TAF inside a vehicle.

**Keywords:** Trust Assessment Framework · Trust Models · In-vehicle Security.

## 1 Introduction

Vehicles on the road today use data from various sensing devices to perform many safety-critical functions, such as Adaptive Cruise Control (ACC). ACC can use data from the radar, lidar, global navigation satellite system (GNSS), and camera devices [12] to calculate the distance between the ego vehicle and its neighboring vehicles. The ACC function can then compare this distance to a predetermined threshold and instruct the vehicle to decelerate or accelerate accordingly, depending on whether the distance is lower or higher. The integrity

* This work is fully funded by the HORIZON CONNECT project under EU grant agreement no. 101069688. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

of the data the ACC or any other safety-critical function uses must not be compromised for the vehicle not to endanger the safety of its passengers or any other traffic participant. In the case of ACC, input data whose integrity has been compromised could lead to the vehicle calculating an erroneous distance and accelerating to the point of causing a crash with the vehicle in front of it. To detect such incidents, vehicles already have reactive security mechanisms such as the intrusion detection system (IDS) in place. Early detection can speed up development and distribution of security updates. However, there are many different detection techniques, and no single IDS can detect a full range of attacks on in-vehicle networks and is especially vulnerable to unknown attacks [1]. This renders the IDS unreliable in ensuring that a safety-critical function does not use data whose integrity has been compromised. Moreover, even when an IDS detects an intrusion, it is challenging to reliably assess the consequences of that intrusion on the rest of the system and, consequently, what appropriate reaction strategies should be. This leaves us in dire need of a solution that can correctly assess the trustworthiness of data given the detailed knowledge of the complex in-vehicle System-of-Systems which the data flow through.

For this purpose, we build upon our prior work in [7] to propose an architecture for the in-vehicle Trust Assessment Framework (TAF), envisioned as a system component whose primary function is to assess whether certain data is trustworthy with respect to its integrity not being compromised. Such trust assessment is made per the request of an application running on the vehicle computer before it needs to execute a safety-critical function. Knowing whether it can trust the data which it needs to run a function allows the application to either use the data or to run appropriate response strategies, e.g., discarding the data, running additional plausibility checks, and even modifying the application to work with less trustworthy data with larger safety margins [13]. The trustworthiness assessment in the TAF is always done at run-time. However, the objects for which trust is assessed can either be known at design time (and not change beyond that) or can dynamically change at run-time. As part of this paper, we focus on the former. For this purpose, we built a trust model for ACC that represents the in-vehicle components involved in producing, processing, or relaying the data needed for the ACC function. We show how various pieces of evidence, ranging from MBD to hardware security mechanisms, can be used to assess the trustworthiness of these components, ultimately enabling us to assess the trustworthiness of all input data. Finally, we present two versions of TAF implementation inside a vehicle: centralized and decentralized.

## 2 Previous Work

### 2.1 Background on Subjective Logic

Subjective Logic (SL) is a mathematical framework that has recently gained prominence due to its ability to deal with the degree of uncertainty of propositions [6]. SL inherently allows uncertainty representation as part of the fundamental building block of SL called *subjective opinion* ($\omega_X^A$).

**Definition 1 (Binomial Subjective Opinion [6]).** *A binomial opinion about the truth of value of $X$ is the ordered quadruplet $\omega_X^A = (b_X^A, d_X^A, u_X^A, a_X^A)$, where the additivity requirement: $b_X^A + d_X^A + u_X^A = 1$ is satisfied, and $b_X^A$ is the belief mass, $d_X^A$ is the disbelief mass, $u_X^A$ is the uncertainty mass, and $a_X^A$ is the base rate. $X$ indicates the target variable or proposition to which the opinion applies, and $A$ indicates the agent who holds the opinion.*

The formalism of SL enables reasoning about the uncertainties and introduces the concept of subjective trust network (STN) that we use as part of this paper. An STN is a directed graph that represents *referral trust* and *functional (or belief) trust* relationships from agents, via other agents to target variables, where each relationship is expressed as a subjective opinion [6].

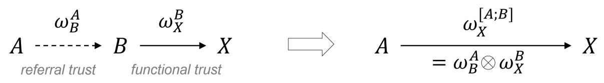


Fig. 1: STN, referral and functional trust, updated from [6].

In Figure 1, we show a simple example of an STN graph (on the left hand side of the figure), where the opinion $\omega_X^{[A;B]}$ is calculated as follows:

$$\omega_X^{[A;B]} = \omega_B^A \otimes \omega_X^B, \tag{1}$$

The symbol $\otimes$ depicts a trust discounting operator for deriving trust from transitive trust paths as shown in the figure.

In this paper, the propositions (e. g., $X$ in Figure 1) that we are interested in assessing are always in relation to data. Consequently, for the purpose of this paper, we modify Eq. 1, to include the notion of *scope*:

$$\omega_X^{[A;B],S} = \omega_{B,S}^A \otimes \omega_X^B,$$

where the subjective opinion of agent $A$ on agent $B$ w.r.t. scope $S$ ($\omega_{B,S}^A$) is discounted by the subjective opinion of agent $B$ on the proposition $X$. Since this paper focuses on integrity, we define the scope here as $S$ = *node's trustworthiness not to compromise the integrity of $X$ data type.*

## 2.2 Related efforts

There are many security mechanisms in in-vehicle networks, for example, gateway firewalls, that protect vehicles from external or internal attacks, ensuring the system's integrity. These gateways prevent unintended messages from being sent from one network to another, such as when an unauthorized application sends messages with compromised data to security-critical networks [8].

In addition to gateway firewalls, intrusion detection systems (IDS) are used in in-vehicle networks. The IDS detects attacks by analyzing messages sent through the network based on certain characteristics [5].

At the electronic control unit (ECU) level, security mechanisms are integrated to protect the integrity of the data. In AUTOSAR standard, for example, the ECUs can use Message Authentication Codes (MAC) and Freshness Verification for CAN messages. In this way, an ECU can ensure that it is communicating with the correct ECU and that the message has not been changed during transmission [3].

In addition to security mechanisms to protect communication, there are security mechanisms that protect the application running in the ECU from other, possibly malicious, applications in the same ECU. One example of this is running applications in isolated virtual machines. Here, a hypervisor hosts multiple virtual machines, each running a separate operating system on which different applications are executed. In this way, the applications are isolated from each other [11].

The mechanisms mentioned above are only a small set of possible security mechanisms that could be implemented in vehicles to protect the integrity of in-vehicle components and the data. Since a vehicle is a complex System-of-Systems, different security mechanisms can be implemented in each component. Therefore, if data flows through several components, the destination component does not know which other components were involved in processing and relaying the data and whether the data's integrity has been compromised. Thus, it is very difficult for the destination component to analyze whether the integrity of the data was always protected. This is where our Trust Assessment Framework comes into play, as it considers the trustworthiness of all components involved in producing, processing, or relaying the data.

## 3 Trust Assessment Framework (TAF)

Our Trust Assessment Framework (TAF) is envisioned as a system component whose primary function is to assess the level of trustworthiness of a certain entity for a specific trust goal. The entity can either be a *node* within the system architecture (which can be a physical node like an ECU connected to a network or a logical node like a library within a firmware) or *data* that is being exchanged between nodes (like a position within a CAM message sent from one to another vehicle). The trust goal we focus on in this paper is *integrity*.

If the respective entity is data, then the trust goal of trustworthiness assessment is *data integrity*. In this case, the TAF is assessing how trustworthy it is that the data has not been compromised. In the case of ACC, compromised data could be incorrect GNSS position data due to a spoofing attack. If the entity whose level of trustworthiness is being assessed is a node, then the trust goal of trustworthiness assessment is *node integrity*, i. e., how trustworthy it is that the node integrity has not been compromised. Note that a node whose integrity has not been compromised is defined as a node whose functionalities and handling of data do not affect the integrity of that data. The higher the trustworthiness of either data or node is, the more likely it is that the TAF will decide to trust that the integrity of that data or node has not been compromised.

The request for the TAF to assess trustworthiness, referred to as *Trustworthiness Assessment Request (TAR)*, comes from an application able to run many different functions. TAR is sent when an application seeks to run a specific safety-critical function, such as Adaptive Cruise Control (ACC). A TAR is the application's main input to the TAF which expects the TAF to output a *Trust Decision (TD)*. The TD is a binary decision whether or not an entity is trustworthy and is obtained after the *Actual Trustworthiness Level (ATL)* is compared to a *Required Trustworthiness Level (RTL)*. The ATL, $\omega_X^{TAF}$, is a subjective opinion (see Section 2.1), and it represents the current level of trustworthiness of an entity $X$ as assessed by the TAF. The RTL represents the required level of trustworthiness of an entity $X$ and is calculated at design time. The RTL can also be calculated as a subjective opinion or have a completely different syntax and semantics. A TAR must include a set of RTL values for each data whose trustworthiness is being assessed.

For the TAF to make a TD, it requires a vast knowledge of the system architecture, including which nodes are involved in which data flow. This knowledge is embedded in trust models stored in the TAF at design time. Each trust model represents a single function run by an application. The application's TAR must specify which safety-critical function is queued to be run by providing an ID of the matching trust model. This lets the TAF analyze an appropriate trust model to decide which data's trustworthiness needs to be assessed for the function to be run.

The architecture of the TAF is given in Figure 2. As can be seen from the architecture, the components that the TAF consists of are:

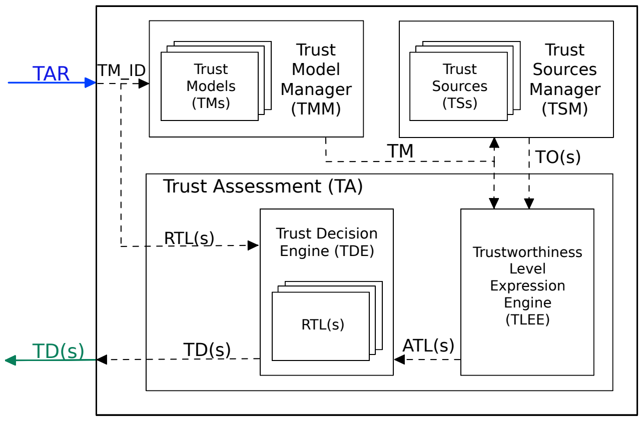

TAR
TM_ID
Trust Models (TMs)
Trust Model Manager (TMM)
Trust Sources (TSs)
Trust Sources Manager (TSM)
TM
TO(s)
Trust Assessment (TA)
RTL(s)
Trust Decision Engine (TDE)
RTL(s)
Trustworthiness Level Expression Engine (TLEE)
TD(s)
TD(s)
ATL(s)

Fig. 2: Trust Assessment Framework (TAF) Architecture

**Trust Model Manager (TMM)** is responsible for selecting the appropriate trust model (TM) based on the TAR received from the application. TMM selects an appropriate model from a set of TMs created and stored during design time. A TM is always matched to a specific function being run and informs the TAF about which data the application needs to run that function, i.e., which data the TAF needs to output the ATL for. TMM also informs the TAF about which trust sources should be used to assess the ATL. What trust models are and how they are created is explained in Section 4.
**Trust Sources Manager (TSM)** is responsible for storing a pre-defined list of all possible trust sources (TSs) that are used for assessing an entity's trustworthiness. The list of trust sources is defined at design time. Based on the trust model, the TSM will collect evidence from appropriate trust sources during run-time and analyze this evidence to derive *Trust Opinions (TOs)*. What these trust sources are and how they are used to assess trustworthiness is explained in Section 5.
**Trust Assessment (TA)** consists of two main parts: the *Trustworthiness Level Expression Engine (TLEE)* and the *Trust Decision Engine (TDE)*. The TLEE is responsible for processing the trust model and input from the Trust Sources Manager to produce an Actual Trustworthiness Level (ATL) by using appropriate Subjective Logic operators. We use the underlying formalism of Subjective Logic to evaluate Trust Models [6], and our TLEE implements this approach. Details of how exactly this is done are out of the scope of this paper, and in future work, we will present this evaluation process in more detail. The TDE stores RTLs when it receives them from the application. Moreover, the TDE receives the ATL from the TLEE, and it compares the ATL with the RTL to obtain a Trust Decision on a certain entity. The Trust Assessment process is explained in more detail in Section 6.

## 4 Trust Models and Trust Model Management

Trust models are core building blocks for the trust assessment within the TAF. They are defined at design time and are always matched to the specific function under consideration that runs as part of an in-vehicle application. After the TAF receives the TAR during run-time, the TMM selects the trust model that matches the function the application wants to run.

We design the trust model based on a *components diagram* that captures the components of the system and the components' relationships necessary to realize that concrete function, for example, the ACC. The components' relationships depict the data flow through the in-vehicle architecture. In Figure 3, we represent the data flow by colored arrows in the component diagram. This was done because we realized assessing the trustworthiness of every component that produces, processes, or relays on data was necessary. A graphical representation of a trust model for an ACC function is shown in Figure 3.

The vertices in the trust model are called *trust objects*—entities that assess trustworthiness or for which trustworthiness is assessed, based on which trust

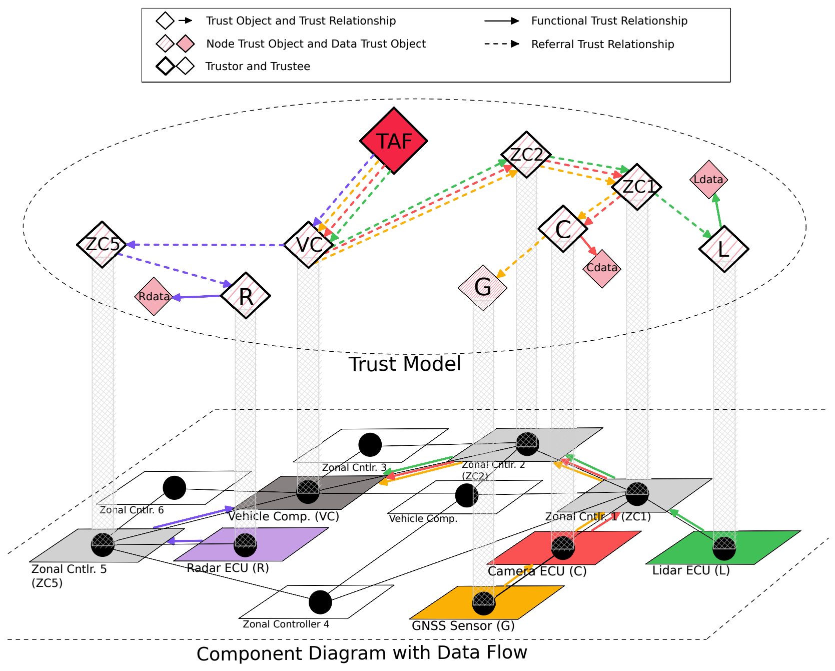


Fig. 3: ACC Component Diagram (including Data Flow) and its Trust Model

relationships are built. Trust objects can represent both: nodes (e. g., vehicle computer, $VC$) or data (e. g., camera feed, $Cdata$). Hence, the trust objects are identified based on 1) the components (i. e., nodes) from the component diagram that are needed for the fulfillment of the specific function under consideration and 2) the concrete propositions that we are interested in assessing—in relation to data. Concretely, a proposition describes the fulfillment of a trust goal of data, and propositions are always in the leaves of the trust model. The trust goal we focus on as part of this paper is integrity. Furthermore, a trust relationship is a directional relationship between two trust objects: trustor and trustee. The trustor is a source trust object as part of a trust relationship for which trustworthiness is assessed (one who trusts). The trustor can only be a node. Whereas the trustee is the sink trust object as part of a trust relationship for which trustworthiness is assessed (one who is trusted). The trustee can be both: 1) a node (in a referral trust), or 2) data (in a functional trust), see Figure 3. In different trust relationships, the same trust objects can be both trustors or trustees, e. g., trust object $ZC2$ in Figure 3.

The trust model comprises various trust relationships among different trust objects. If each trust relationship in the trust model is expressed as a subjective opinion, then our trust model becomes a Subjective Trust Network (STN), see Section 2.1. In this context, we refer to these subjective opinions per trust relationship as *Trust Opinions* (note that the TMM does not build Trust Opinions, the TSM does this).

Moreover, what is not visually captured in Figure 3 is that each trust object contains information about which other trust objects it assesses trustworthiness of, if any, and which exact trust sources are used to assess this trustworthiness and how. Note, however, that trust objects only store this information. Choosing the specific trust sources for different trust relationships is part of the design of the trust model. The derivation of the numerical value for the Trust Opinion, based on the trust sources, is performed by the TSM at run-time. This will be further discussed in Section 5.

## 5 Trust Sources

As described above, a trust relationship is focused on integrity in this paper. Therefore, the trust sources to be selected need to provide evidence for the integrity of trust objects in a trust relationship. The trust sources are specified at design time for individual trust relationships and are linked to the trust model. These trust sources can be determined by experts or based on a risk assessment analysis of the trust objects. At run-time, the Trust Sources Manager analyzes the specified trust sources, retrieves evidence from the components represented by trust objects, and then uses this evidence to assess Trust Opinions. We divide the trust sources into four categories. Depending on the specific trust object in a trust model, not all categories may be relevant. The first three categories are related to communication, operating system (OS), and application processing the data, as all these have access to the data and could therefore compromise it. The trust sources in these categories are predominantly security mechanisms. In addition to security mechanisms, the actual behavior of the component represented by the trust object could be analyzed to obtain further evidence and is thus the fourth category of trust sources. Some examples of possible trust sources in each category are listed below. This list is not exhaustive; additional trust sources could be added to each category.

### 5.1 Trust in communication

*Integrity protected communication.* Some communication protocols provide integrity protection mechanisms. Such mechanisms allow one to detect if a message was changed during transmission. Therefore, using a communication protocol with integrity protection provides evidence for integrity.

*Hardware security mechanisms.* Hardware security mechanisms such as hardware security modules (HSM) or trusted platform modules (TPM) are capable, among other things, of managing and storing cryptographic keys and executing cryptographic functions without the application having access to the required keys. This, for example, makes it more difficult for attackers to obtain access to keys relevant to encrypting or signing messages. Therefore, if hardware security mechanisms are used, it becomes more difficult to impersonate another node in a network which provides evidence for integrity [2].

### 5.2 Trust in OS/Firmware

*Secure boot.* Secure boot verifies that a malicious actor has not altered or tampered with the software components relevant to the boot process, such as the boot loader or OS. Thus, manipulated or unauthenticated software can be detected, and the trustworthiness of the OS/Firmware can be adapted accordingly [9].

*Known OS-vulnerabilities.* Based on the OS/Firmware version of the node and a vulnerability database, it can be determined whether there are known vulnerabilities in the node. The vulnerabilities usually also contain a risk value, which shows how critical the vulnerability is. Based on that, the trustworthiness of the OS/Firmware can be adjusted. A similar approach could be used as a trust source for application vulnerabilities.

### 5.3 Trust in application

*Run-time operational assurance.* Run-time operational assurance mechanisms, such as control-flow integrity, detect the applications' operations to be modified by an attacker during run-time. This makes realizing a broad range of attacks more difficult, which can be reflected in the trustworthiness level of the application that uses the input data [4].

*Trusted Execution Environment (TEE).* The TEE provides a trusted environment where data can be stored and code can be executed. Access to the data in a TEE is controlled in order to protect the data from attacks outside the TEE. The code can also not be seen or modified by entities outside the TEE. This makes it much more difficult to compromise the application [9].

### 5.4 Trust built on behavior

*Plausibility check.* Misbehavior detection checks data provided by a node to determine if the node is misbehaving. A plausibility check is one detector of misbehavior detection, which can be used as a trust source. Different approaches are possible to check the plausibility of the data. For example, a vehicle's position could be compared with other inputs, such as a map, to check whether the position is within a road [10].

*Intrusion detection system (IDS).* A network-based IDS monitors a network of systems for malicious activities or suspicious behavior. All malicious activities or behaviors are collected and combined to determine malicious nodes within the network that would make the corresponding node less trustworthy.

### 5.5 Deriving Trust Opinions

We developed two approaches for deriving Trust Opinions. The first approach is linear, where each trust source has an equivalent impact on the resulting Trust Opinion. Thus, depending on whether the trust source provides positive or negative evidence, the belief or disbelief and the uncertainty change by the same amount. The second approach is the weighted approach, where a weight is specified for each trust source in the trust model. Experts could define such weights or they could be derived based on a risk assessment. These weights are used for deriving a Trust Opinion and indicate how much the belief, disbelief, and uncertainty should change. How the weighted and linear approaches differ in the quality of the derived Trust Opinion and which other approaches could be used for deriving Trust Opinions is part of future work.

To illustrate the derivation of Trust Opinions, for simplicity, the linear approach is used to quantify one of the trust relationships between the Vehicle Computer and Zonal Controller 2 (ZC2) seen in Figure 3. We focus on the trust relationship whose scope is ZC2's trustworthiness not to compromise the integrity of camera data. Trust sources specified for this trust relationship could be integrity-protected communication, hardware security mechanism, secure boot, run-time operational assurance, and IDS.

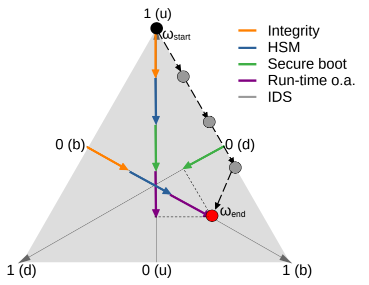


Fig. 4: Derivation of a Trust Opinion based on the linear approach

Since the linear approach is used and there are five designated trust sources, each source has a weight of 0.2, meaning that the belief (b), disbelief (d), and uncertainty (u) will be increased or decreased by 0.2. The Trust Opinion starts at $\omega^{VC}_{ZC,S}(b = 0, d = 0, u = 1)$ because no evidence about the trust entity exists at the beginning. Due to the presence of integrity-protected communication, the uncertainty is decreased by 0.2, and the belief is increased by 0.2. As a hardware security mechanism (TPM) protects the keys used for communication protection, uncertainty is reduced by 0.2, and belief is again increased by 0.2. Since the secure boot failed, there is negative evidence, the uncertainty is reduced by 0.2, and the disbelief is increased by 0.2. In addition, the run-time operational assurance has no detections; hence, the uncertainty is reduced, and the belief is increased by 0.2. Finally, since no IDS exists in the system, no further evidence can be provided, and the final Trust Opinion is $\omega^{VC}_{ZC,S}(b = 0.6, d = 0.2, u = 0.2)$. Figure 4 shows how much each trust source changes the Trust Opinion and how the final Trust Opinion is derived from the initial Trust Opinion.

## 6 In-Vehicle Trust Assessment

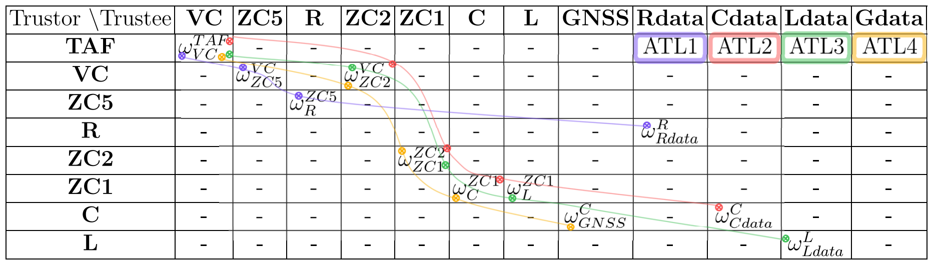
In this section, we exemplify the usage of our TAF for an ACC function which shows how the trustworthiness is assessed at run-time. We investigate how the TAF fits into the overall vehicle architecture and interacts with other in-vehicle components to produce the required output. As part of this paper, we propose two different versions of the TAF: centralized and decentralized, and we explain both of these versions in more detail later in this section.

ACC function in today's vehicles can rely on many in-vehicle data such as the lidar, radar, GNSS, and camera data [12]. The level of trustworthiness of all of this data needs to be assessed by the TAF before the application can use the data for the ACC function. For this to happen, the engineer must create an appropriate trust model for the ACC function at design time. In Figure 3 in Section 4, we showed an example of the ACC trust model. In this example, there is a trust object for every component that either produces the data (GNSS sensor $G$, Lidar ECU $L$, Camera ECU $C$, and Radar ECU $R$), relays the data (Zonal Controllers: $ZC1$, $ZC2$, $ZC5$) or processes the data (Vehicle Computer $VC$). There is also a trust object for every data created by a component with enough computing power to host a TAF ($Ldata$, $Cdata$, $Rdata$). In this case, only the GNSS sensor is assumed not to be able to host a TAF and the trustworthiness of its data will be judged by the trustworthiness of the GNSS sensor itself. As previously explained in Section 4, after the initial trust model has been created, the model will be expanded to include a list of appropriate trust sources which need to be checked during run-time for each Trust Opinion assessment. With this added, the model is stored in the TMM on the TAF.

During run-time, upon receiving the TAR from the ACC application, the TMM forwards the matching trust model to the TA, whose first task will be to figure out which ATLs are required by the application. This is done by identifying the leaves of the trust model, which would, in this case, be the following trust objects: $Rdata$, $Cdata$, $Ldata$, and $G$. The required ATLs are as follows: $ATL1 = \omega^{TAF}_{Rdata}$, $ATL2 = \omega^{TAF}_{Cdata}$, $ATL3 = \omega^{TAF}_{Ldata}$, and $ATL4 = \omega^{TAF}_{G}$. The second task of the TA is to identify the paths in the trust model which lead from the TAF to the leaves. As can be seen in Figure 5, the purple-colored path in the example ACC trust model connects the following trust objects: $TAF \rightarrow VC \rightarrow ZC5 \rightarrow R \rightarrow Rdata$. In this path, the following TOs need to be assessed to obtain $ATL1$: $\omega^{TAF}_{VC,S1}$, $\omega^{VC}_{ZC5,S1}$, $\omega^{ZC5}_{R,S1}$ and $\omega^{R}_{Rdata}$, where $S1 =$ *[node's trustworthiness not to compromise the integrity of R.data]*. The TSM assesses these TOs based on the lists of necessary trust sources extracted from the trust model. The TSM first collects the evidence for the necessary trust sources and then uses this evidence to assess TOs. Once they are returned to the TLEE, the TLEE uses the following formula (refer to Section 2) to produce $ATL1$:

$$ATL1 = \omega^{TAF}_{Rdata} = \omega^{TAF}_{VC,S1} \otimes \omega^{VC}_{ZC5,S1} \otimes \omega^{ZC5}_{R,S1} \otimes \omega^{R}_{Rdata} \tag{2}$$

An ATL is then forwarded to the TDE, where it is compared to an RTL to obtain a Trust Decision. Future work will focus on how a meaningful Trust Decision is formed, primarily as it could be related to comparing ATL and RTL with different semantics. In addition, it is necessary to investigate how the TAF would be implemented into the overall vehicle architecture and how it would interact with other components to collect evidence or distribute the trustworthiness assessment to different nodes. For this, we propose a centralized and a decentralized TAF.



| Trustor \Trustee | **VC** | **ZC5** | **R** | **ZC2** | **ZC1** | **C** | **L** | **GNSS** | **Rdata** | **Cdata** | **Ldata** | **Gdata** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **TAF** | $\omega_{VC}^{TAF}$ | - | - | - | - | - | - | - | ATL1 | ATL2 | ATL3 | ATL4 |
| **VC** | - | $\omega_{ZC5}^{VC}$ | - | $\omega_{ZC2}^{VC}$ | - | - | - | - | - | - | - | - |
| **ZC5** | - | - | $\omega_{R}^{ZC5}$ | - | - | - | - | - | - | - | - | - |
| **R** | - | - | - | - | - | - | - | - | $\omega_{Rdata}^{R}$ | - | - | - |
| **ZC2** | - | - | - | - | $\omega_{ZC1}^{ZC2}$ | - | - | - | - | - | - | - |
| **ZC1** | - | - | - | - | - | $\omega_{C}^{ZC1}$ | $\omega_{L}^{ZC1}$ | - | - | - | - | - |
| **C** | - | - | - | - | - | - | - | $\omega_{GNSS}^{C}$ | - | $\omega_{Cdata}^{C}$ | - | - |
| **L** | - | - | - | - | - | - | - | - | - | - | $\omega_{Ldata}^{L}$ | - |

Fig. 5: ACC Trust Model Paths

## 6.1 Centralized TAF

A centralized TAF implies a single central TAF inside a vehicle, most likely deployed on a dedicated component and running inside a Trusted Execution Environment (TEE). The central TAF receives a function-specific TAR from an application and issues what we term **Requests for Evidence (RFEs)** that are sent to various other vehicle components. RFEs are issued to collect evidence at the central TAF that will allow the TAF to assess the trustworthiness for all trust relationships inside the trust model. RFEs contain the identifier of the component they are intended for and the list of the evidence needed. Which components the requests are sent to depends on the trust model of the function. Each of these components needs to be able to process the RFE, collect the requested evidence, and send it back to the TAF, but also to forward the RFE along when the requested evidence is supposed to come from another component down the line. The components are expected to be able to host an Evidence Manager (EM), which would perform this functionality. The evidence becomes the input the Trust Sources Manager will use to populate its trust sources and derive the Trust Opinions. Note that, in this case, even though the trust model may require an opinion of, e.g., the Zonal Controller 5 on the Radar ECU, $\omega_R^{ZC5}$, it will be the central TAF who will assess this opinion and any other opinion needed by the trust model. Hence, even though the semantics of opinions in the trust model say otherwise, the central TAF assesses all of the opinions, ultimately producing the desired TD. An example sequence diagram of a centralized TAF for $ATL1 = \omega_{Rdata}^{TAF}$ is shown in Figure 6. As can be seen from the figure, upon receiving the TAR, the TAF will issue multiple RFEs, which will then be distributed between other components. The components collect the evidence locally and promptly send it back through the network to the TAF. This approach does not require full functionality of the TAF to be implemented on each in-vehicle component. However, only the Evidence Manager is needed, which reduces the computing load on individual components. On the other hand, with this approach, the central TAF has to perform lots of calculations independently, which may increase the total processing time.

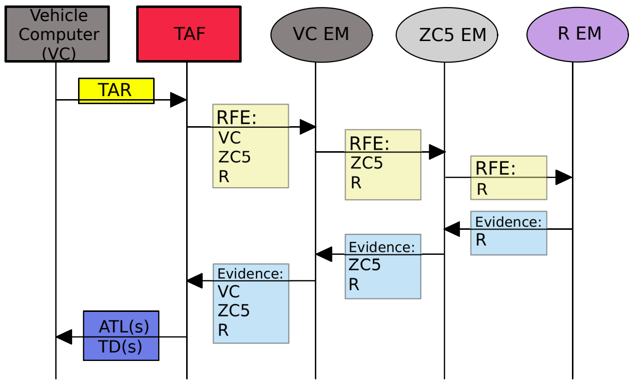


Fig. 6: Centralized TAF - Flow of RFEs and Evidence

## 6.2 Decentralized TAF

A decentralized TAF approach implies several TAFs inside the vehicle, all working distributedly. Similar to the centralized approach, there is a main TAF, most likely sitting on a dedicated component and inside a Trusted Execution Environment. Moreover, there are local TAFs inside various in-vehicle components whose range of functionality is similar to the main TAF. As with the centralized approach, the main TAF receives a function-specific TAR from an application. In this case, however, the TAF does not issue RFEs, but rather the **Requests for Opinions (RFOs)**. This way, the main TAF distributes the load by requesting local TAFs to collect evidence and produce opinions. An example sequence diagram of a decentralized TAF for $ATL1 = \omega^{TAF}_{Rdata}$ is shown in Figure 7. In this case, upon processing the TAR, the TAF realizes that one of the ATLs required by the application is $\omega^{TAF}_{Rdata}$. The TAF knows that, in the context of preserving the integrity of $Ldata$, it has a referral trust relationship with the Vehicle Computer, which processes that data once it receives it from the data source. As a result, the TAF will send an RFO to the Vehicle Computer requesting an opinion of the VC on $Rdata$, $\omega^{VC}_{Rdata}$. Similarly, the vehicle computer, equipped with its own local TAF, will receive the RFO, analyze the same trust model which was already stored inside the local TAF, and "pass down the baton" by sending an RFO to the next component in the line, and in this case, a request for $\omega^{ZC5}_{Rdata}$. This happens for all of the involved components until the data source is reached, and the source's opinion on its data is requested, $\omega^{R}_{Rdata}$. The data source's local TAF will then build an opinion on the data by using appropriate trust sources and forward this opinion with evidence of its own trustworthiness. Once the ZC5 TAF receives the opinion and the evidence, it will process the evidence to assess its own opinion on the source, $\omega^{ZC5}_{R}$. Next, the ZC5 TAF will use the Subjective Logic's trust discounting operator to discount $\omega^{ZC5}_{R}$ with $\omega^{R}_{Rdata}$, ultimately obtaining the requested $\omega^{ZC5}_{Rdata}$ which it finally forwards along. The exact process occurs on every component which follows, including the main

TAF, which discounts its own evidence-based opinion on the vehicle computer, $\omega_{VC}^{TAF}$, with the VC's opinion on the radar data, $\omega_{Rdata}^{VC}$, to obtain the ATL. This approach requires full functionality of the TAF to be implemented on most vehicle components, increasing the load of individual components but potentially reducing the overall time needed to produce a requested TD. We recognize that some components inside the vehicle may not have the computing capability to host a full TAF and could not produce the opinion of its data. In this case, the next component in the line able to host a TAF will assess the opinion on the data source itself, and this opinion will represent the trustworthiness of the data. Here, the data is as trustworthy as the device that creates it.

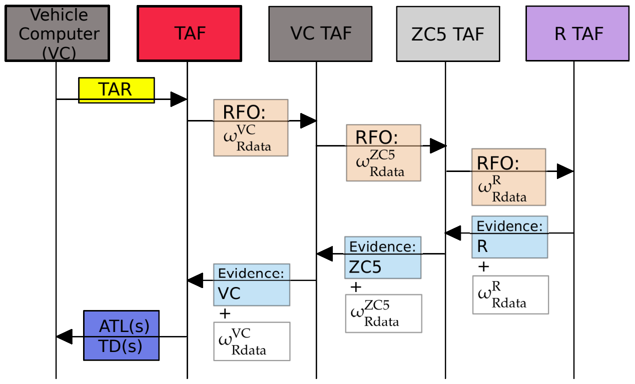


Fig. 7: Decentralized TAF - Flow of RFOs, Evidence, and Opinions

## 7 Conclusion

In this paper, we presented the Trust Assessment Framework envisioned as an in-vehicle system component whose task is to assess the trustworthiness of data input to safety-critical functions running as part of in-vehicle applications. We proposed two versions of the Trust Assessment Framework to be deployed inside a vehicle: centralized and decentralized. Both versions have advantages and disadvantages; hence, their detailed comparison, security, and timing performance will be part of our future work. Moreover, we created a trust model representing in-vehicle components that could compromise the integrity of data used as input to the Adaptive Cruise Control function. We showed how the Trust Assessment Framework uses this model with various types of trust sources to assess the trustworthiness of ACC input data. In future work, we plan to build a prototype Trust Assessment Framework and investigate different methods of deriving Trust Opinions.

## References

1. Al-Jarrah, O.Y., Maple, C., Dianati, M., Oxtoby, D., Mouzakitis, A.: Intrusion detection systems for intra-vehicle networks: A review. IEEE Access **7**, 21266–21289 (2019)
2. Apvrille, L., El Khayari, R., Henniger, O., Roudier, Y., Schweppe, H., Seudié, H., Weyl, B., Wolf, M.: Secure automotive on-board electronics network architecture. In: FISITA 2010 world automotive congress, Budapest, Hungary. vol. 8 (2010)
3. AUTOSAR: Classic Platform AUTOSAR, https://www.autosar.org/standards/classic-platform
4. Burow, N., Carr, S.A., Nash, J., Larsen, P., Franz, M., Brunthaler, S., Payer, M.: Control-Flow Integrity: Precision, Security, and Performance. ACM Computing Surveys **50**(1), 16:1–16:33 (Apr 2017), https://dl.acm.org/doi/10.1145/3054924
5. Hoppe, T., Kiltz, S., Dittmann, J.: Applying intrusion detection to automotive IT-early insights and remaining challenges. Journal of Information Assurance and Security (JIAS) **4**, 226–235 (Jan 2009)
6. Jøsang, A.: Subjective Logic. Artificial Intelligence: Foundations, Theory, and Algorithms, Springer International Publishing, Cham (2016), http://link.springer.com/10.1007/978-3-319-42337-1
7. Kargl, F., Trkulja, N., Hermann, A., Sommer, F., Ramon Ferraz de Lucena, A., Kiening, A., Japs, S.: Securing cooperative intersection management through subjective trust networks. In: 2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring). pp. 1–7 (2023)
8. Le, V.H., den Hartog, J., Zannone, N.: Security and privacy for innovative automotive applications: A survey. Computer Communications **132**, 17–41 (Nov 2018), https://www.sciencedirect.com/science/article/pii/S014036641731174X
9. Sabt, M., Achemlal, M., Bouabdallah, A.: Trusted Execution Environment: What It is, and What It is Not. In: 2015 IEEE Trustcom/BigDataSE/ISPA. vol. 1, pp. 57–64 (Aug 2015)
10. So, S., Sharma, P., Petit, J.: Integrating Plausibility Checks and Machine Learning for Misbehavior Detection in VANET. In: 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA). pp. 564–571 (Dec 2018)
11. Stumpf, F., Sit, F.I., Meves, C., Weyl, B., Wolf, M.: A Security Architecture for Multipurpose ECUs in Vehicles. In: 25. VDI/VW-Gemeinschaftstagung: Automotive Security. Ingolstadt, Germany (Oct 2009)
12. Winner, H., Hakuli, S., Lotz, F., Singer, C.: Handbook of Driver Assistance Systems: Basic Information, Components and Systems for Active Safety and Comfort. Springer International Publishing (Aug 2015), google-Books-ID: Qu3IwAEACAAJ
13. Wolf, M., Willecke, A., Müller, J.C., Garlichs, K., Griebel, T., Wolf, L., Buchholz, M., Dietmayer, K., van der Heijden, R.W., Kargl, F.: Securing CACC: Strategies for Mitigating Data Injection Attacks. In: 2020 IEEE Vehicular Networking Conference (VNC). pp. 1–7 (Dec 2020), iSSN: 2157-9865