Conceptual Framework for Adaptive Safety in Autonomous Ecosystems

David Halasz[a] and Barbora Buhnova[b]
Faculty of Informatics, Masaryk University, Brno, Czech Republic

Keywords: Autonomous Collaborative Ecosystems, Adaptive Safety, Software Architecture, Trust, Security, Autonomous Vehicles, Smart Agents, Digital Twins.

Abstract: The dynamic collaboration among hyper-connected Autonomous Systems promotes their evolution towards Autonomous Ecosystems. In order to maintain the safety of such structures, it is essential to ensure that there is a certain level of understanding of the present and future behavior of individual systems in these ecosystems. Adaptive Safety is a promising direction to control access to features between cooperating systems. However, it requires information about its collaborators within the environment. Digital Twins could be used to predict possible future behavior of a system. This paper introduces a conceptual framework for Adaptive Safety that is being triggered based on the trust score computed from the predictive simulation of Digital Twins, which we suggest to use in Autonomous Ecosystems to load and safely execute third-party Smart Agents. By quantifying trust towards the agent and combining it with a decision tree, we leverage this as a deciding factor to conceal or expose certain features among collaborating systems.

1 INTRODUCTION

Traditionally Autonomous Cyber-Physical Systems leverage computer networking to share information among each other. This communication is inevitably forming a dynamic Autonomous Ecosystem of heterogeneous member systems (Cioroaica et al., 2019a). As the complexity of these Systems of Systems is growing, more sophisticated ways of information sharing are required for coordination, to the degree where an execution of a Smart Agent received from the central authority or other collaborators might be needed for effective synchronization and collaboration (Capilla et al., 2021). However, executing third-party software in any environment raises safety concerns.

Although there are technologies that allow isolated execution of untrusted software (Sahoo et al., 2010; Greamo and Ghosh, 2011), a full isolation would render the benefits of these Smart Agent insignificant over other technologies. Already existing pass-through solutions often rely on human intervention and static configuration, which is not applicable in a constantly changing autonomous environment. To overcome this issue, a more sophisticated architecture is necessary that is capable of an adaptive permission handling of pass-through configurations. In other words, the access of a third-party Smart Agent to the various platform services provided by an Autonomous System should be dynamically changing based on the actual context.

1.1 Motivating Scenarios

In any city around the world, local authorities might state rules and regulations for vehicles. These local rules and regulations are usually displayed in horizontal or vertical road signs and it is up to the drivers to respect them. Although there are solutions for Autonomous Vehicles to recognize and respect these signs, there might be more sophisticated solutions for enforcement. Using a medium intended for humans (road signs) to communicate with machines might be an unnecessary extra step. Furthermore, there might be more complicated dynamic rules that are specific to Autonomous Vehicles that can be only represented in executable machine code. To support this, the possibility to safely load and execute third-party Smart Agent into an existing Autonomous System would be necessary.

Autonomous Driving in a Smart City. Consider a scenario of an Autonomous Vehicle entering the premises of a city where speed limits are dynamically

[a] https://orcid.org/0000-0002-0393-7857
[b] https://orcid.org/0000-0003-4205-101X

Halasz, D. and Buhnova, B.
Conceptual Framework for Adaptive Safety in Autonomous Ecosystems.
DOI: 10.5220/0012086600003538
In *Proceedings of the 18th International Conference on Software Technologies (ICSOFT 2023)*, pages 393-403
ISBN: 978-989-758-665-1; ISSN: 2184-2833

enforced based on the location, time of the day and current density of traffic. In order to reduce the traffic on certain spots, there are also zones with the minimum speed that are intended to empty critical intersections faster. Let's imagine that the city requires all Autonomous Vehicles joining its ecosystem to load and execute a Smart Agent implementing these dynamic rules of traffic. Even though the Smart Agent has been certified by regulatory organizations, executing it in a privileged mode where it has unlimited access to all the features of the vehicle carries some risks. A simple binary decision about providing access is not sufficient to ensure that both the vehicle and the ecosystem maintain their safety requirements.

Vehicle Platooning. A different scenario might happen on a highway, where vehicles can move in platoons(Axelsson, 2016). These ad-hoc groups of vehicles moving closely together with a matching speed allow them to utilize aerodynamic properties to reduce their energy consumption. In a centralized scenario, the Smart Agent may be created in advance by the highway authorities and distributed using Road Side Units. However, a more dynamic and decentralized approach is also possible when a vehicle already contains an agent for platooning and shares it with its neighbors. The advantage of this scenario is a lower reliance on a Road Side Unit and the fact that the life cycle of the Smart Agent in the ecosystem can be controlled by the actual platoon. On the other hand, a vehicle with malicious intent could easily disrupt the whole ecosystem and cause damage to other members.

1.2 Contribution of the Paper

In this paper, we envision a conceptual framework for trust-based adaptive safety assurance in autonomous ecosystems, which is tailored (though not limited) to the motivating context of Autonomous Vehicles. The proposed framework is based on the vision that the potentially untrusted Smart Agents (discussed in the scenarios) should be executed in an environment where safety is ensured based on trust. By quantifying the trust towards an agent, the framework suggests to adjust the agent's reach to various features of the Autonomous System (e.g., vehicle).

As the trust score may vary over time, the framework supports continuous real-time reevaluation of the trust quantification and adaptation to it. The key component of the proposed framework is thus the trust quantification process, which is fed from two sources: (1) reputation engine and (2) predictive simulation of the Smart Agent conducted on a "virtual copy" of the agent, i.e. a Digital Twin (Rosen et al., 2015), intended for describing the behavior of the agent in a simulated environment. The framework can then use the Digital Twin to simulate possible future scenarios happening to the autonomous vehicle, and to perform runtime compliance checking (Iqbal and Buhnova, 2022) of the Digital Twin with the actual real-world behaviour of the Smart Agent, to assess the accuracy of the Digital Twin as well as the trustworthiness of the Smart Agent.

1.3 Paper Structure

The structure of the paper is as follows. After the discussion of the background in terms of trust management and safety assurance in Section 2, we discuss the related work in Section 3, highlighting the concepts of safety assurance in autonomous systems and trust management in software systems in general. The envisioned conceptual framework is presented in Section 4 and later evaluated on a demonstration scenario in Section 5. The paper concludes with discussion in Section 6 and conclusion in Section 7.

2 BACKGROUND

To lay down the foundations for trust-based adaptive safety in autonomous ecosystems, we first explain the essentials of autonomous ecosystems and define the key concepts in trust management, which are mainly the representation of trust and trust-formation components. Then, we look into the basics of safety assurance and its understanding in the context of autonomous ecosystems.

2.1 Autonomous Ecosystems

With the expanding scope of today's system, characterized by hyper-connectivity and dynamic runtime reconfiguration, adapting system behaviour to effectively collaborate with its surroundings, we are witnessing the transitioning from standalone systems and systems-of-systems to dynamic software ecosystems (Capilla et al., 2021). Within dynamic software ecosystems, the individual systems and other entities (including the Smart Agents inside these systems) are engaged in mutual interactions, which might be driven by collaborative, competitive or even malicious goals (Cioroaica et al., 2021). This stimulates the need of the individual collaborators to be able to distinguish among the entities they can and cannot trust, forming social relationships to support their interests and needs (Sagar et al., 2022).

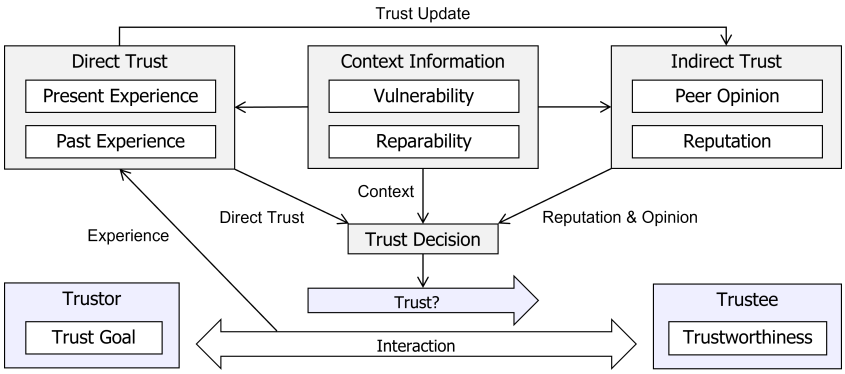

Figure 1: Trust Management Components (Buhnova, 2023).

2.2 Trust Management

Every branch of science has its own definition of trust (Cho et al., 2015). The most applicable ones that match our understanding are coming from human-related branches (Rotter, 1980; Lahno, 2001; Mayer et al., 1995) that can be projected to the context of autonomous systems via understanding trust as *"the attitude or belief of an agent (trustor) to achieve a specific goal in interaction with another agent (trustee) under uncertainty and vulnerability"* (Buhnova, 2023).

2.3 Representation of Trust

Finding the right representation for the depth of trust might be contextual, with each domain having its own preference. The majority of the research focuses on a binary understanding (Cho et al., 2015), i.e. to trust or not to trust. However, the large distance between these two extremes renders this approach risky in case of misclassification. As we argued in our previous work (Halasz, 2022; Halasz and Buhnova, 2022), by interpreting trust as a quantifiable value, e.g. a percentage, we can smoothen out the extremes by statistically reducing the distance between a wrongly assessed and the actual trust. Without specifying its actual representation in detail, we refer to this representation as the Trust Score.

2.4 Trust Formation Components

The processes of trust formation by a trustor towards a trustee are typically fed from two main sources (see Figure 1), the direct and indirect trust (Buhnova, 2023). *Direct Trust* represents an individual judgment by the trustor based on its direct interaction with the trustee. In this sense, it is based on local and context-specific experience and direct observation by the trustor. *Indirect Trust* is, on the other hand, learned from the environment, in terms of the reputation of the trustee. Specifically, it can come from an authority managing the reputation of the trustee within the ecosystem, or be formed by propagated opinions and recommendations of trusted peers of the trustor.

2.5 Safety in Autonomous Ecosystems

Similarly to trust, safety can be understood in a variety of ways depending on its context. According to our analysis, the following definitions are the most appropriate for our context: *"the ability of a distributed application and its parts to continue operating in a safe manner during and after a transformation"* (McKinley et al., 2004) and the *"avoidance of hazards to the physical environment"* (Banerjee et al., 2012). Safety in the domain of Autonomous Ecosystems is significantly broader than safety in Autonomous Systems (Ramos et al., 2019). The dynamic collaboration of member systems creates a larger collaborative entity that has a different interpretation of safety that might even violate safety requirements for some of its members in the interest of the ecosystem as a whole.

2.6 Safety Assurance

Ensuring safety in reasonably complex systems can be mainly covered during design-time considerations with a thorough testing, compliance checking and certification process before it is ready for the end-users. Even though these systems contain run-time dynamic solutions for safety assurance, their flexibility is limited and can not always adapt to uncertain situations. Therefore, when working with more complex systems of ecosystems, a different approach is necessary. Instead of a static design-time certification, a dynamic run-time certification is necessary to ensure that a system behaves safely even in previously not covered sit-

uations (Kusnirakova and Buhnova, 2023; Bakirtzis et al., 2022).

3 RELATED WORK

While research work on safety in the wide context of autonomous ecosystems is still in its inception (see Section 3.1), substantial attention is being targeted towards the safety of autonomous vehicles (see Section 3.2). Besides, related work can also be identified in the direction of isolated execution of untrusted software, used to ensure safety (see Section 3.3), and coordination of agents in autonomous ecosystems for safe collaboration (see Section 3.4).

3.1 Safety in Autonomous Ecosystems

There has been interesting relevant research done to ensure safety and security in the particular areas of autonomous systems. The biggest two gaps identified in this body of work are uncertainties in modeling and accomplishing higher goals through cooperation and collaboration (Jahan et al., 2019). The concept of self-adaptation offers a solution of dealing with scenarios that are uncertain and not known during design time. Technologies like MAPE-K (Arcaini et al., 2015) or aCLFs (Taylor and Ames, 2020) can be utilized to achieve security and safety via self-adaptation. Unfortunately, these methods do not always scale well as solutions to the magnitude of Autonomous Ecosystems, e.g. there is no straightforward solution to distribute a MAPE-K feedback loop among members of an ecosystem.

3.2 Autonomous Vehicles Safety

Unquestionably, the domain of Autonomous Vehicles has been the focus of study in the field of Autonomous Systems in recent years. Research in the field of safety in this area is largely concerned with communication security, emergency response plans in case of failure or attack (Cui et al., 2019; Bouchelaghem et al., 2020), dynamic construction of vehicle convoys, i.e. platooning (Axelsson, 2016), and collision avoidance techniques (Li et al., 2021). Although all these research topics are in the domain of safety, they mainly focus on individual vehicles and only very few of them have a broader focus that can be scaled up to our vision of ecosystems. Both individual and group coordination by simple information sharing has its limits and heavily relies on keeping all the software in every actor of an ecosystem up to date to maintain compatibility.

3.3 Isolated Execution of Untrusted Software

Technologies like virtualization (Sahoo et al., 2010) and sandboxing (Greamo and Ghosh, 2011) are the industry-standard, among others, for isolating untrusted code from the rest of a system. These solutions usually implement pass-through (Waldspurger and Rosenblum, 2012; Yang et al., 2014) mechanisms to allow the isolated software to access certain services. Mobile application platforms (Laverty et al., 2011) heavily build on such technologies to provide a pass-through mechanism with user-configurable permission control (Felt et al., 2011). Ensuring safety in Cyber-Physical Systems was also inspired by these technologies, Bak et al. proposed an idea (Bak et al., 2011) of combining a sandbox with the Simplex Architecture (Seto et al., 1998; Sha, 2001) to switch between the execution of an unverified and verified component. The main issue with these solutions is that they only provide a binary solution to safety, i.e. a complete switch to a safety mode, disallowing access to the questionable agent. In complex ecosystems, such solutions might be insufficient, given the expected granularity of certainty about the component trustworthiness (Halasz, 2022; Halasz and Buhnova, 2022).

3.4 Smart Agents in Autonomous Ecosystems

Executing Smart Agents to coordinate members of an ecosystem of Autonomous Vehicles is already a part of the AUTOSAR standard (Fürst et al., 2009). However, ensuring that a malicious agent can not do harm in the ecosystem by leveraging trust as a decision factor is a relatively new idea (Cioroaica et al., 2019b). Assessing the trustworthiness could be possible by using predictive simulation on a Digital Twin of the Smart Agent (see Figure 2) (Cioroaica et al., 2020a). This approach already has some promising early results, and has certain aspects (namely the aspects of

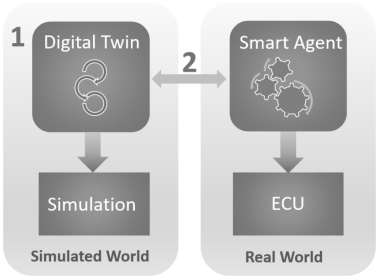

Figure 2: Predictive simulation using a Digital Twin (Cioroaica et al., 2019b).

trust goals) covered by a preliminary reference architecture (Cioroaica et al., 2020b). Unfortunately, the research is still in its initial phases and there is no detailed information about how it would ensure safety, leaving it as a suggestion for future work.

4 CONCEPTUAL FRAMEWORK

In this paper, we envision a conceptual framework for adaptive safety in Autonomous Ecosystems under the execution of possibly untrustworthy Smart Agents. The framework is inspired by the mechanisms used in mobile application platforms, like Android, which create an abstraction over a device and use a permission-based access control around their services (Laverty et al., 2011; Neuner et al., 2014; Felt et al., 2011). However, instead of a binary and user-controlled permission scheme, we propose an adaptive non-binary solution (Halasz, 2022; Halasz and Buhnova, 2022) that can adjust the permissions in real time, based on a continuously calculated trust score (Cioroaica et al., 2020a; Cioroaica et al., 2019b). Our goal with this framework is to provide a sandbox for safe execution of third-party Smart Agents in Cyber-Physical Systems (Cioroaica et al., 2020b; Pivoto et al., 2021). The architectural view of this framework is in Figure 3, combining the mechanisms of both direct and indirect trust, as discussed in Section 2.

4.1 Agent Execution Environment

The core of the sandbox depicted in Figure 3 is an Agent Execution Environment that loads and runs third-party Smart Agents. As stated before, this is inspired by mobile application platforms (Laverty et al., 2011). Any loaded agent should be in a predefined executable format that the environment understands and supports. An Autonomous Vehicle should provide a platform with which the framework can interface. Even though standardization efforts for such platforms exist, in the context of safety, we can not simply rely on the existence of a universal platform that works consistently across vendors. In order to overcome vendor-specific differences, a Virtual Vehicle Interface is necessary. This interface would provide an abstraction using drivers to any standardized vehicle platform, which shall allow the framework to support any Autonomous Vehicle in the future.

4.2 Gatekeeper

The entry point into this framework is a Gatekeeper responsible for receiving a bundle of a Smart Agent with its Digital Twin. Having a single point of entry for any incoming data reduces the possible attack surface of the overall ecosystem. Its first task is to verify the digital signature of the bundle and make sure that both the Smart Agent and its Digital Twin are in the right format. Additionally, some malicious instructions can be detected by heuristic scanning (Bazrafshan et al., 2013) and other static analysis (Gosain and Sharma, 2015) techniques. A separate part of the Gatekeeper is a Reputation Engine capable of sending and receiving information about the reputation of a Smart Agent, feeding the indirect-trust information into the framework. This reputation information is being consumed inside the framework by multiple internal components.

4.3 Digital Twin Verifier

Before allowing the Smart Agent or its Digital Twin to be executed, the Digital Twin Verifier runs a set of selected simulations on the Digital Twin (Boschert and Rosen, 2016). The initial reputation received by the Reputation Engine can affect the selection process. A bundle with a high reputation that has already been functioning in other Autonomous Vehicles can be verified faster by skipping some steps of the verification process. On the other hand, a bundle with no history and hence no/low reputation needs to go through a set of more thorough checks. At the end of the verification process, the Digital Twin is either passed further to the Digital Twin Simulator or the framework rejects the whole bundle. Regardless of this decision, the verification results are passed to the Trust Aggregator for further processing.

4.4 Digital Twin Simulator

Within the framework, a successfully verified Digital Twin gets loaded into a predictive simulation (Cioroaica et al., 2019b; Cioroaica et al., 2020a) environment inside the Digital Twin Simulator. The difference between the Verifier and the simulator is that the latter works with a context received via a read-only clone of the Virtual Vehicle Interface. This ensures that the Digital Twin has no way to alter the behavior of the vehicle under any circumstances, which is crucial for safety control. While the simulation relies on real-world data, it is ahead of the real world by a short time difference to predict the behavior of a Smart Agent. Similarly to the Verifier, any irregu-

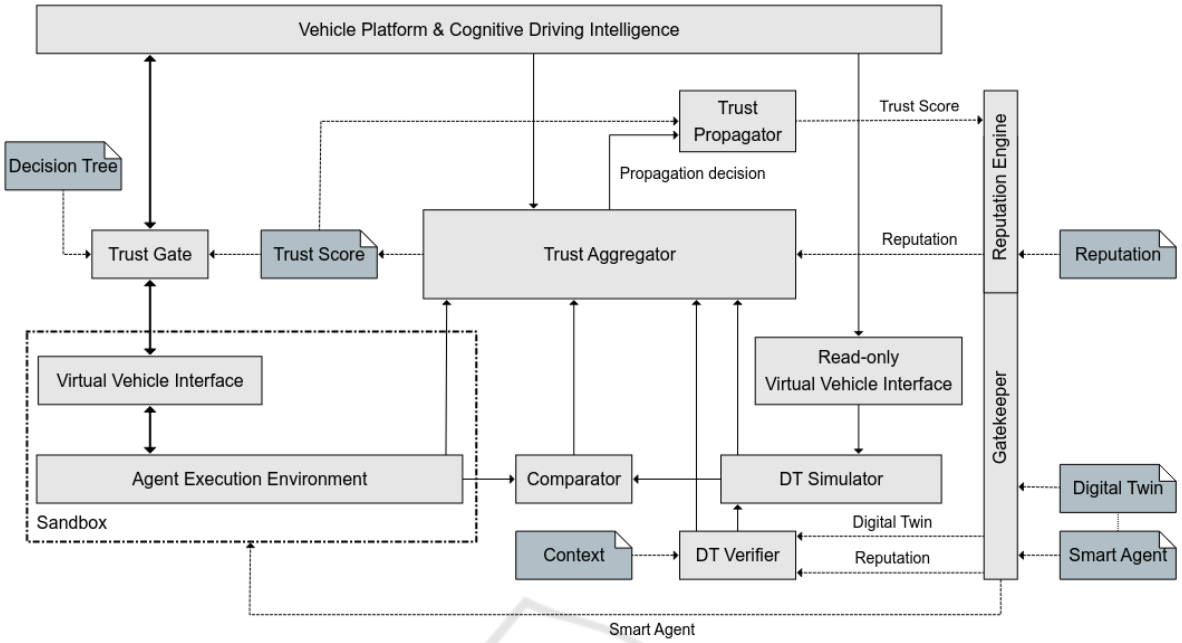

Figure 3: Architectural View of the Conceptual Framework.

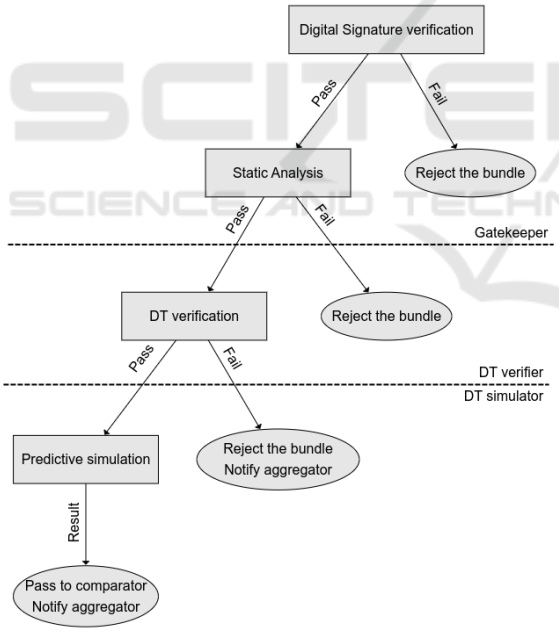

Figure 4: Lifespan of the Digital Twin.

larities during the simulation are passed further to the Trust Aggregator.

4.5 Comparator

Between the Digital Twin Verifier and the Agent Execution Environment, a Comparator balances out the time difference by acting as a buffer. Its main task is to compare the results between these two components and also to act as a continuous compliance checker. It can detect any deviation between the behavior of the Agent and the Digital Twin, which it can then report to the Trust Aggregator.

4.6 Trust Aggregator

Any behavioral information about external reputation, Digital Twin verification, simulation, and compliance checking gets collected by the Trust Aggregator. As its name suggests, this component continuously aggregates all this information into a Trust Score. This is a quantified subjective representation of the trustworthiness of the Smart Agent. The actual representation of the Trust Score depends on the chosen aggregation method (Cho et al., 2015), e.g. a percentage or even a vector of multiple percentages that represent various aspects of the Autonomous Vehicle. If there is a significant change in the Trust Score, the aggregator can decide to trigger the Trust Propagator to pass the Trust Score to the Reputation Engine and the outside world (Govindan and Mohapatra, 2011).

4.7 Trust Gate

The final and most crucial component of the framework is the Trust Gate between the Virtual Vehicle Interface and the Vehicle Platform. Based on a predefined Decision Tree (Halasz, 2022; Halasz and Buhnova, 2022) and the continuously changing Trust

Score, it dynamically controls the access to features provided by the Vehicle Platform. This tree is heavily specific to the given Autonomous Vehicle and assigns the exposure or concealment of vehicle features to Trust Scores.

The life cycle of the Digital Twin in terms of its handling by the framework is illustrated in form of a decision tree in Figure 4.

5 VALIDATION ON SELECTED SCENARIOS

In this section, to better illustrate the proposed concept, we demonstrate the envisioned framework on a two scenarios. One in a Smart City with Intelligent Transportation and another ono on a highway with ad-hoc Vehicle Platooning.

5.1 Smart City

Consider a Smart City using a Smart Agent bundled with its Digital Twin to enforce traffic rules. This bundle is digitally signed by the local department of transportation and it contains instructions for speed limits and minimum speed requirements in certain areas of the city. In order to enforce these rules, it requires access to the following platform services: location and positioning, speed limiter, emergency braking, cruise control and lane switching control. Propagation is solved by wireless networks created by roadside units on the edges of the city premises.

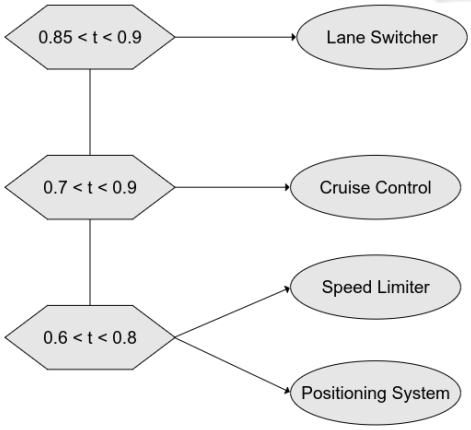

Figure 5: Simplified Decision Tree for Demonstration.

Imagine an Autonomous Vehicle implementing our framework that is moving towards this city. Its simplified Decision Tree is displayed in Figure 5, the Trust Score is represented with a number from the interval $< 0, 1 >$ and the chosen aggregation method is just an arithmetic average of all inputs. The vehicle enters the premises of the city and automatically receives the bundle from the nearest wireless network. The static analysis scanners in the *Gatekeeper* component find everything correct, meanwhile the *Trust Aggregator* receives a score of, let's say, 0.4 from the *Reputation Engine* which means that other vehicles had poor past experience with this bundle. Based on this low score, the *Digital Twin Verifier* runs all the predefined simulation scenarios and verifies that all of them pass. Therefore, the *Trust Aggregator* receives a score of 1.0, calculates an initial Trust Score of, say, 0.7 and passes it further to the *Trust Gate*. According to the vehicle-specific Decision Tree, the *Trust Gate* allows initial access to the positioning system and the speed limiter features.

At this point the *Digital Twin Simulator* starts testing possible future scenarios using the real-world context coming from the *Read-only Virtual Vehicle Interface*. Meanwhile, the Smart Agent is loaded into the sandbox and starts interfacing with the initially permitted two platform services. As the vehicle moves through a zone with limited speed, the *Comparator* verifies that the Digital Twin is compliant with the actual behavior of the Smart Agent as both behave the same way in case of a speed limit. This causes a gradual increase in the Trust Score to, say, 0.8 and further to 0.85 over time. This higher score triggers the *Trust Gate* to expose the cruise control service to the sandbox, which can be used to set a minimum speed for the vehicle.

When the vehicle enters a two-lane road where both lanes have a minimum speed requirement as visualized in Figure 6, the predictive simulation already verified that the exposure of the cruise control service did not cause any irregularities in the behavior of the Digital Twin. In the meantime, the Smart Agent initiates the cruise control to match the minimum speed of the current lane where the vehicle is present. Due to the fact that there is no deviation detected by the *Comparator*, the Trust Score is increased to 0.9, exposing the lane-switching feature to the sandbox.

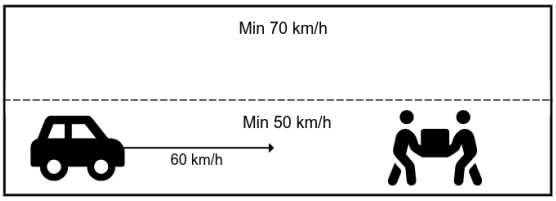

Figure 6: Two-lane road with minimum speeds and an obstacle.

As the vehicle moves further on the road, its sensors detect that there is an obstacle further in front

of it. The *Digital Twin Simulator* runs a predictive check of what would happen if the vehicle switches lanes. Unfortunately, there would be a mismatch between the embedded collision avoidance system and the Smart Agent as the latter would want to keep the vehicle in the lane to match the speed, but that would cause the vehicle to risk collision with the obstacle. This unexpected result causes the *Trust Aggregator* to reduce the Trust Score to 0.8, which disables the access of the sandbox to lane switching. As the agent can no longer interfere with the embedded collision avoidance system, the vehicle changes lanes for the time being to pass the obstacle.

It is possible, that the Smart Agent provided by the city is outdated and did not get proper integration testing with modern collision avoidance systems. This could explain the poor experience of other cars that resulted in a 0.4 reputation score. On the other hand, the speed-limiting feature works safely and the adaptive mechanism in the framework did not limit its reach.

5.2 Vehicle Platooning

Consider a single autonomous vehicle *A* on a three-lane highway that is capable of running Smart Agents. It has no software support for vehicle platooning, but it is equipped with our framework. Gradually, vehicles *B*, *C* and *D* implementing the same framework (see Figure 7) join the highway, from which only vehicle *C* contains a Smart Agent supporting Vehicle Platooning.

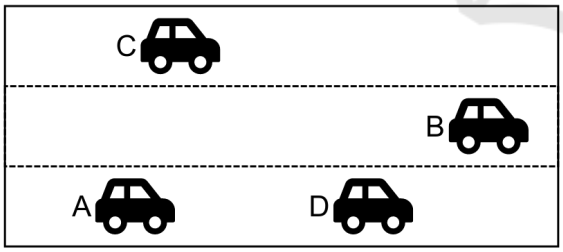

Figure 7: Three-lane highway with four autonomous vehicles not cooperating.

After some time passes to collect enough data from its surroundings, vehicle *C* evaluates that if all the vehicles join into a single lane and match their speed and distance, they can improve their joint aerodynamic resistance and reduce overall fuel consumption. The vehicle uses an ad-hoc wireless network to propagate its bundle containing the Smart Agent and its Digital Twin responsible for vehicle platooning to all three neighboring vehicles. As the distance and the network quality between these vehicles is different, furthermore, they are manufactured by different vendors, the time between propagation and adoption of this bundle might differ for each vehicle. The first to receive the bundle is vehicle *A* as it is the closest to vehicle *C*. There might be reputation information propagated alongside with the bundle, however, as it is coming from the issuer of the Smart Agent, the *Digital Twin Verifier* is instructed to run all the tests on the Digital Twin before allowing it to enter the *Digital Twin Simulator*. Similarly to the previous example, upon successful verification, the Smart Agent and the Digital Twin are loaded into their intended environment and the *Trust Aggregator* initially assigns them a *Trust Score* of 0.75. This allows the Smart Agent to access the cruise control feature of the *Vehicle Platform* via the *Trust Gate* and results in vehicle *A* maintaining the optimal distance for platooning and matching the speed of vehicle *C*, while still staying in different lanes.

Vehicles *B* and *D* are also going through the same process, however, they can additionally consume the *Trust Score* calculated by vehicle *A* and take it into account when calculating their own *Trust Score*. This results in a faster verification process as certain steps already done by vehicle *A* can be omitted and a slightly higher score of 0.8 which allows these two vehicles to match their speed and distance required for the platoon. As the Smart Agent in all three new adopters has been proven as safe, the *Trust Score* in all of them can gradually increase above the threshold of 0.85, which enables the agent to access the lane switcher feature. By exposing this feature, the Smart Agent in each vehicle can instruct them to move to the first lane and form the desired platoon (see Figure 8).

As all vehicles have a good experience with the platooning agent, its *Trust Score* in each vehicle can grow further to 0.90. This triggers the *Trust Gate* of each vehicle to allow higher speeds via the *Vehicle Platform*. If the predictive simulation in vehicle *D* would detect behavior that is different from what happens in reality, the *Trust Score* is lowered back to 0.85. This score automatically revokes the access of the *Smart Agent* to higher speeds, and the vehicle can

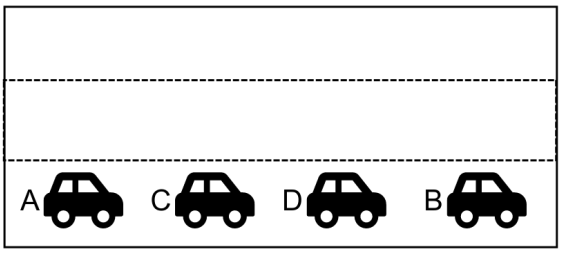

Figure 8: Three-lane highway with four autonomous vehicles forming a single platoon.

either leave the platoon by switching lanes or (if they support adaptive cruise control) instruct the other cars that it will gradually decrease its speed. Meanwhile, the newly calculated score is propagated to the other vehicles that can adjust their *Trust Gate* accordingly. As long as the *Trust Score* towards the platooning *Smart Agent* stays high in each vehicle, the platoon will continue to operate, even if the initiator vehicle *C* leaves the highway. It is even possible that the platoon itself can outlive all its original members.

6 DISCUSSION

The proposed conceptual framework is still an initial step in our vision towards a solution for ensuring safety in Autonomous Ecosystems by leveraging trust. Some key components of the framework are not fully specified yet or have been largely simplified in order to facilitate the demonstration. Furthermore, it is still not clear if it is feasible to use a single universal trust aggregation method and what should be the actual representation of the Trust Score. Our plan for the future is to find an answer to these questions and give a more detailed description of the framework that could be mapped to real-world scenarios.

6.1 Generalizability of the Concept

Although the framework has been designed to execute third-party Smart Agents, we see potential to utilize the concept in different contexts of Autonomous Cyber-Physical Systems. Both outside the vehicular ecosystems and outside the execution of malicious software. For instance, a sensor attached to a system (e.g., Autonomous Vehicle) might produce invalid measurements and damage either the system itself or its surroundings. We believe that a similar framework could overcome such problems by quantifying to which degree the system can trust its individual sensors. Based on the Trust Score, the measured values can be either ignored or adjusted. This solution could bring a certain level of fault tolerance into the system that was not available before.

6.2 Proof-of-Concept Implementation

This architecture could be implemented for distributed and collaborative software systems in Cloud or Edge Computing environments. Container orchestration platforms such as Kubernetes already implement a Role-based Access Control (RBAC) service that governs the access for consumers of service accounts to certain resources(Binnie and McCune, 2021). By implementing a similar framework on top of the Kubernetes platform, it would be possible to provide an analogous Trust-based access control (TBAC) service that would govern resource access based on the trust between these consumers of service accounts. Individual pods could also tap into this service and leverage its features to restrict or grant access to other pods interacting with it.

Research around the use of edge computing in the supporting architecture for autonomous vehicles (Tang et al., 2021; Sandu and Susnea, 2021; Tian et al., 2022) already leverages Kubernetes and similar platforms. There is already an approach to turn parked Autonomous Vehicles currently not in use to edge devices (Nguyen et al., 2022). This suggests a possible future where Autonomous Vehicles would use Kubernetes as the base of their software stack. This would open the possibility of a reference implementation of our framework purely using Kubernetes resources. In this case it would have a minimal difference from the previously described TBAC service.

6.3 Flipping the Concept

The solution proposed in this paper supports the increase of safety in an overall collaborative Autonomous Ecosystem via the support for the safe execution of a Smart Agent by a system acting as a host for the agent. Another interesting opportunity for safety assurance of a collaborative ecosystem that supports the execution of a Smart Agent in an individual ecosystem member is to use the Smart Agents to control the safe behaviour of the host system to prevent it from engaging in harmful actions towards its collaborators (e.g., a vehicle hitting a break when in the middle of a platoon). This essentially means flipping the idea. Besides looking at a potentially untrustworthy Smart Agent executing in a trustworthy host system, we would like to investigate the support for a trustworthy Smart Agent executing in a potentially untrustworthy host system (and controlling it towards its safer behaviour). In our future work, we would like to elaborate on this idea further.

7 CONCLUSION

The unstoppable evolution of Autonomous Ecosystems requires a new approach to ensure the safe operation of both the individual member systems and the ecosystem as a whole. Coordinating a set of heterogeneous Autonomous Systems (e.g., vehicles) by simple information sharing is not sufficient to handle dynamic context changes and uncertain situations.

A more effective solution could be to employ Smart Agents distributed and deployed at the collaborating systems to facilitate their coordination. However, this inherently carries new types of threats to safety that need to be addressed. Our approach builds on assessing the trustworthiness of these Smart Agents by using predictive simulation and leverages the resulting Trust Score as a decision factor in what system services should be exposed to them. In this paper, we present a conceptual framework implementing this idea, tailored to the context of Autonomous Vehicles, and demonstrate it on a simple scenario. The next steps in our research are to create a reference implementation of the proposed framework and validate it with the help of our industrial partners.

ACKNOWLEDGEMENTS

This research was supported by ERDF "CyberSecurity, CyberCrime and Critical Information Infrastructures Center of Excellence" (No. CZ.02.1.01/0.0/0.0/16_019/0000822).

REFERENCES

Arcaini, P., Riccobene, E., and Scandurra, P. (2015). Modeling and analyzing mape-k feedback loops for self-adaptation. In *2015 IEEE/ACM 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, pages 13–23.

Axelsson, J. (2016). Safety in vehicle platooning: A systematic literature review. *IEEE Transactions on Intelligent Transportation Systems*, 18:1–13.

Bak, S., Manamcheri, K., Mitra, S., and Caccamo, M. (2011). Sandboxing controllers for cyber-physical systems. In *2011 IEEE/ACM Second International Conference on Cyber-Physical Systems*, pages 3–12.

Bakirtzis, G., Carr, S., Danks, D., and Topcu, U. (2022). Dynamic certification for autonomous systems.

Banerjee, A., Venkatasubramanian, K. K., Mukherjee, T., and Gupta, S. K. S. (2012). Ensuring safety, security, and sustainability of mission-critical cyber–physical systems. *Proceedings of the IEEE*, 100(1):283–299.

Bazrafshan, Z., Hashemi, H., Fard, S. M. H., and Hamzeh, A. (2013). A survey on heuristic malware detection techniques. In *The 5th Conference on Information and Knowledge Technology*, pages 113–120. IEEE.

Binnie, C. and McCune, R. (2021). Kubernetes authorizationwith rbac.

Boschert, S. and Rosen, R. (2016). Digital twin—the simulation aspect. *Mechatronic futures: Challenges and solutions for mechatronic systems and their designers*, pages 59–74.

Bouchelaghem, S., Bouabdallah, A., and Omar, M. (2020). Autonomous Vehicle Security: Literature Review of Real Attack Experiments. In *The 15th International Conference on Risks and Security of Internet and Systems*, Paris, France.

Buhnova, B. (2023). Trust management in the Internet of Everything. In *Proceedings of the 16th European Conference on Software Architecture-Companion Volume*, pages 1–13. To appear in Springer. Preprint at http://arxiv.org/abs/2212.14688.

Capilla, R., Cioroaica, E., Buhnova, B., and Bosch, J. (2021). On autonomous dynamic software ecosystems. *IEEE Transactions on Engineering Management*, pages 1–15.

Cho, J.-H., Chan, K., and Adali, S. (2015). A survey on trust modeling. *ACM Comput. Surv.*, 48(2).

Cioroaica, E., Buhnova, B., Kuhn, T., and Schneider, D. (2020a). Building trust in the untrustable. In *2020 IEEE/ACM 42nd International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*, pages 21–24. IEEE.

Cioroaica, E., Chren, S., Buhnova, B., Kuhn, T., and Dimitrov, D. (2019a). Towards creation of a reference architecture for trust-based digital ecosystems. In *Proceedings of the 13th European Conference on Software Architecture-Volume 2*, pages 273–276.

Cioroaica, E., Chren, S., Buhnova, B., Kuhn, T., and Dimitrov, D. (2020b). Reference architecture for trust-based digital ecosystems. In *2020 IEEE International Conference on Software Architecture Companion (ICSA-C)*, pages 266–273. IEEE.

Cioroaica, E., Kuhn, T., and Buhnova, B. (2019b). (do not) trust in ecosystems. In *2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, pages 9–12. IEEE.

Cioroaica, E., Purohit, A., Buhnova, B., and Schneider, D. (2021). Goals within trust-based digital ecosystems. In *2021 IEEE/ACM Joint 9th International Workshop on Software Engineering for Systems-of-Systems and 15th Workshop on Distributed Software Development, Software Ecosystems and Systems-of-Systems (SESoS/WDES)*, pages 1–7. IEEE.

Cui, J., Liew, L. S., Sabaliauskaite, G., and Zhou, F. (2019). A review on safety failures, security attacks, and available countermeasures for autonomous vehicles. *Ad Hoc Networks*, 90:101823. Recent advances on security and privacy in Intelligent Transportation Systems.

Felt, A. P., Chin, E., Hanna, S., Song, D., and Wagner, D. (2011). Android permissions demystified. In *Proceedings of the 18th ACM conference on Computer and communications security*, pages 627–638.

Fürst, S., Mössinger, J., Bunzel, S., Weber, T., Kirschke-Biller, F., Heitkämper, P., Kinkelin, G., Nishikawa, K., and Lange, K. (2009). Autosar–a worldwide standard is on the road. In *14th International VDI Congress Electronic Systems for Vehicles, Baden-Baden*, volume 62, page 5. Citeseer.

Gosain, A. and Sharma, G. (2015). Static analysis: A survey of techniques and tools. In *Intelligent Computing and Applications: Proceedings of the International Conference on ICA, 22-24 December 2014*, pages 581–591. Springer.

Govindan, K. and Mohapatra, P. (2011). Trust computations and trust dynamics in mobile adhoc networks: A survey. *IEEE Communications Surveys & Tutorials*, 14(2):279–298.

Greamo, C. and Ghosh, A. (2011). Sandboxing and virtualization: Modern tools for combating malware. *IEEE Security Privacy*, 9(2):79–82.

Halasz, D. (2022). From systems to ecosystems: Rethinking adaptive safety. In *17th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS '22)*. IEEE.

Halasz, D. and Buhnova, B. (2022). Rethinking safety in autonomous ecosystems. In *The 17th Conference on Computer Science and Intelligence Systems*, pages 81–87. IEEE.

Iqbal, D. and Buhnova, B. (2022). Model-based approach for building trust in autonomous drones through digital twins. In *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE.

Jahan, F., Sun, W., Niyaz, Q., and Alam, M. (2019). Security modeling of autonomous systems: A survey. *ACM Comput. Surv.*, 52(5).

Kusnirakova, D. and Buhnova, B. (2023). Rethinking certification for higher trust and ethical safeguarding of autonomous systems. In *Proceedings of the 18th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE)*. Scitepress.

Lahno, B. (2001). On the emotional character of trust. *Ethical theory and moral practice*, 4(2):171–189.

Laverty, J. P., Wood, D. F., Kohun, F. G., and Turchek, J. (2011). Comparative analysis of mobile application development and security models. *Issues in Information Systems*, 12(1):301–312.

Li, G., Yang, Y., Zhang, T., Qu, X., Cao, D., Cheng, B., and Li, K. (2021). Risk assessment based collision avoidance decision-making for autonomous vehicles in multi-scenarios. *Transportation Research Part C: Emerging Technologies*, 122:102820.

Mayer, R. C., Davis, J. H., and Schoorman, F. D. (1995). An integrative model of organizational trust. *The Academy of Management Review*, 20(3):709–734.

McKinley, P., Sadjadi, S., Kasten, E., and Cheng, B. (2004). Composing adaptive software. *Computer*, 37(7):56–64.

Neuner, S., van der Veen, V., Lindorfer, M., Huber, M., Merzdovnik, G., Mulazzani, M., and Weippl, E. R. (2014). Enter sandbox: Android sandbox comparison. *CoRR*, abs/1410.7749.

Nguyen, K., Drew, S., Huang, C., and Zhou, J. (2022). Parked vehicles task offloading in edge computing. *IEEE Access*, 10:41592–41606.

Pivoto, D. G., de Almeida, L. F., da Rosa Righi, R., Rodrigues, J. J., Lugli, A. B., and Alberti, A. M. (2021). Cyber-physical systems architectures for industrial internet of things applications in industry 4.0: A literature review. *Journal of manufacturing systems*, 58:176–192.

Ramos, M. A., Thieme, C. A., Utne, I. B., and Mosleh, A. (2019). Autonomous systems safety–state of the art and challenges. In *Proceedings of the First International Workshop on Autonomous Systems Safety*. NTNU.

Rosen, R., Von Wichert, G., Lo, G., and Bettenhausen, K. D. (2015). About the importance of autonomy and digital twins for the future of manufacturing. *Ifac-papersonline*, 48(3):567–572.

Rotter, J. B. (1980). Interpersonal trust, trustworthiness, and gullibility. *American psychologist*, 35(1):1.

Sagar, S., Mahmood, A., Sheng, Q. Z., Pabani, J. K., and Zhang, W. E. (2022). Understanding the trustworthiness management in the social internet of things: A survey. *arXiv preprint arXiv:2202.03624*.

Sahoo, J., Mohapatra, S., and Lath, R. (2010). Virtualization: A survey on concepts, taxonomy and associated security issues. In *2010 Second International Conference on Computer and Network Technology*, pages 222–226.

Sandu, C. and Susnea, I. (2021). Edge computing for autonomous vehicles-a scoping review. In *2021 20th RoEduNet Conference: Networking in Education and Research (RoEduNet)*, pages 1–5. IEEE.

Seto, D., Krogh, B., Sha, L., and Chutinan, A. (1998). The simplex architecture for safe online control system upgrades. In *Proceedings of the 1998 American Control Conference. ACC (IEEE Cat. No.98CH36207)*, volume 6, pages 3504–3508 vol.6.

Sha, L. (2001). Using simplicity to control complexity. *IEEE Software*, 18(4):20–28.

Tang, S., Chen, B., Iwen, H., Hirsch, J., Fu, S., Yang, Q., Palacharla, P., Wang, N., Wang, X., and Shi, W. (2021). Vecframe: A vehicular edge computing framework for connected autonomous vehicles. In *2021 IEEE International Conference on Edge Computing (EDGE)*, pages 68–77. IEEE.

Taylor, A. J. and Ames, A. D. (2020). Adaptive safety with control barrier functions. In *2020 American Control Conference (ACC)*, pages 1399–1405.

Tian, J.-S., Huang, S.-C., Yang, S.-R., and Lin, P. (2022). Kubernetes edge-powered vision-based navigation assistance system for robotic vehicles. In *2022 International Wireless Communications and Mobile Computing (IWCMC)*, pages 457–462. IEEE.

Waldspurger, C. and Rosenblum, M. (2012). I/o virtualization. *Communications of the ACM*, 55(1):66–73.

Yang, C.-T., Liu, J.-C., Wang, H.-Y., and Hsu, C.-H. (2014). Implementation of gpu virtualization using pci pass-through mechanism. *The Journal of Supercomputing*, 68:183–213.