Quantifying Trust in Autonomous System Under Uncertainties

Raj Gautam Dutta, Xiaolong Guo, and Yier Jin
Department of Electrical and Computer Engineering, University of Central Florida
{rajgautamdutta, guoxiaolong}@knights.ucf.edu, yier.jin@eecs.ucf.edu

***Abstract*—Over the years, autonomous systems have entered almost all the facets of human life. Gradually, higher levels of autonomy are being incorporated into cyber-physical systems (CPS) and Internet-of-things (IoT) devices. However, safety and security has always been a lurking fear behind adoption of autonomous systems such as self-driving vehicles. To address these issues, we develop a framework for quantifying trust in autonomous system. This framework consist of an estimation method, which considers effect of adversarial attacks on sensor measurements. Our estimation algorithm uses a set-membership method during identification of safe states of the system. An important feature of this algorithm is that it can distinguish between adversarial noise and other disturbances. We also verify the autonomous system by first modeling it as networks of priced timed automata (NPTA) with stochastic semantics and then using statistical probabilistic model checking to verify it against probabilistic specifications. The verification process ensures that the autonomous system behave in accordance to safety specifications within a probabilistic threshold. For quantifying trust on the system, we use confidence results provided by the model checking tool. We have demonstrated our approach by using a case study of adaptive cruise control system under sensor spoofing attacks.**

I. Introduction

Automation is being increasingly introduced into every man-made system. Currently, most of the day-to-day applications are *semi-autonomous* as they require human intervention in achieving desired goals. However, the thrust to achieve trustworthy autonomous systems, which can attain goals independently in the presence of significant uncertainties and for long periods of time without any human intervention, has always been enticing. Significant progress has been made in the avenues of both software and hardware for meeting these objectives. Currently, space vehicles are autonomous and there are prototypes of self-driving autonomous vehicles [1]. However, technological challenges still exist and particularly in terms of decision making under uncertainty. In an autonomous system, uncertainties can arise from the operating environment, adversarial attacks, and from within the system. As a result of these ambiguities, human-beings lack trust in these systems and hesitate to use take them for day-to-day use. For an autonomous system, *trust* is defined as the ability of the system to successfully carry out a task, at a particular time, and in a situation characterized by vulnerability and uncertainty [2]. To build trust in an autonomous system, manufacturers have to certify them. Certification is a formal means by which regulators measure expected performance of different components of an autonomous system. A generic autonomous system consist of (i) perception unit, which comprises of sensing system and its data processing software, (ii) control and decision making unit, which decides future actions of the autonomous system, and (iii) execution unit, which carry out actions provided by the control and decision making unit [3]. To measure trust, one has to have quantitative value of following traits of the system (i) performance, which includes competence, reliability, and robustness, (ii) transparency of control and decision-making unit, and (iii) security vulnerabilities [4]. These factors should be measured both in certain and uncertain environments. Thus, design and certification of autonomous systems such as self-driving cars, which can withstand uncertain conditions is of utmost importance.

Robust control algorithms are being used to make an autonomous system resilient against environmental and in-system uncertainties. However, these methods fail to extend toward uncertainties created by adversarial attacks [5]–[12]. As such we use a statistical probabilistic formal verification method for certifying autonomous system as trustworthy and show its resiliency against attacks. In our approach, we assume that the sensors of the system are being compromised by an attacker via spoofing or jamming attacks. As a result, the control and decision making unit of the system get degraded sensor data, which makes the system behave abnormally. We also assume that the control and decision making unit, the actuators, and the diagnostics and fault management sub-system of the execution unit cannot be modified by an attacker. However, these units/sub-systems may be affected by internal and environmental disturbances. With these assumptions, we first estimate a set of safe states for the vehicle. This estimation is done at the diagnostics and fault management sub-system of the execution unit. Then, the safe states are used to monitor the outputs of the control and decision unit. When a suspicious state is detected by this sub-system, it alerts the reactive control sub-system, which is also in the execution unit. This system takes the necessary actions to keep the autonomous system within the realms of safety. As the reactive control sub-system is prone to sensor noise and environmental disturbance, we verify whether it satisfies the safety requirements with certain probability value. Based on the verification results we assign performance measurements to different components of the execution unit of the autonomous system. As 100% certification of autonomous system is nearly impossible, we adopt a probabilistic rating mechanism to assign confidence values to different units of the system. Thus, we quantify trust in the system by adding ratings of all these components. We demonstrate our approach on an autonomous ground vehicle

978-1-5090-1367-8/16/$31.00 ©2016 IEEE

under sensor spoofing attacks.

II. Attack Model

Attack on an autonomous vehicle can be carried out either via physical access [5], [6] or remotely [7]–[11]. In our paper, we consider remote attacks, as attackers getting physical access of the vehicles internal operation may be an infeasible assumption. Remote exploitation is possible via a wide range of attack vectors present in the vehicle such as telematics system, Wi-Fi, Bluetooth, in-car apps, remote keyless entry, Tire Pressure Monitoring System (TPMS), radio data system, and Dedicated Short Range Communications (DSRC), developed for vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2X) communications [11]. These attack vectors uses various wireless communication wavelengths to get access of the in-vehicle network, which comprises of Controller Area Network (CAN) buses, Local Interconnect Network (LIN) buses, Media Oriented System Transports (MOST) buses, and FlexRay buses [13]. As CAN buses communicate with all the Electronic Control Units (ECUs) of the vehicle, most of the attacks are targeted toward it [5], [7], [11]. Control of the CAN bus allows the attacker to take control of most of the functionalities of the vehicle.

Another form of remote attack target data acquisition unit of the vehicle, which comprises of sensors [8]–[10], [12]. An autonomous vehicle has many sensors (radar, lidar, Global Positioning System (GPS), camera, ultrasonic, and inertial measurement unit (IMU)) for collecting data of its internal and external environment [14], [15]. Compromising the data gathered by these sensors can impact the decisions made by the motion control unit of the vehicle.

To degrade sensor data quality, an attacker has to carry out spoofing or jamming attack remotely. Following the attack models described in [8]–[10], [12], we assume that the attacker target the hardware layer (external sensors) and has limited knowledge of sensors firmware or software. We also assume that they have limited resources to carry out the remote spoofing attack. Furthermore, the attack can be mounted on the target vehicle while its stationary or in motion. By carrying out these attacks, the attacker intends to cause physical damage to the vehicle.

Based on the remote spoof attacks carried out in [8]–[10], [12], we can state that the lidar, GPS, radar, camera, and ultrasonic sensors of the vehicle can be compromised. We assume that the sensor fusion unit (shown in Figure 1), which gets processed data of compromised sensors, cannot rectify the malicious data. Other assumptions are that the adversary does not have access to the diagnostic and fault management sub-system of the vehicle and they cannot directly modify the control and decision unit.

III. Related Work

Verification of autonomous ground vehicles has been receiving increasing attention over the years [16]–[19]. Stursberg et al. [16] modeled the cruise control system of vehicle using hybrid automata with nonlinear continuous dynamics and polyhedral guard and invariant sets. Then, they used counterexample-guided verification approach to ensure the cruise control system of two cars in one lane was operating correctly to avoid collision. However, their method could not be scaled to arbitrary number of cars. Loos et al. [17] used quantified hybrid program for modeling distributed car control system, where every car was controlled by adaptive cruise control. They used the automated theorem prover-KeYmaera - to prove that the control model satisfied the collision avoidance protocol for arbitrary number of cars on a street. Wongpiromsarn et al. [18] proposed the framework, called periodically controlled hybrid automata, for modeling control systems in which inputs to actuators were given after a fixed time unit. They verified safety and progress properties of the planner-controller sub-system of an autonomous ground vehicle using sum of square decomposition and semi-definite programming. Althoff et al. [19] used reachability analysis for safety verification of evasive maneuvers of autonomous ground vehicle with constant velocity and under uncertainties.

However, these verification methods do not consider an autonomous ground vehicle under adversarial attacks [16]–[19]. To address this scenario, many attack detection and prevention methods were developed for ground vehicles [20]–[24]. Zheng et al. [20] analyzed security and safety of in-vehicle control system and vehicle-to-vehicle communication network using hybrid modeling, formal verification, and automated synthesis techniques. They considered cooperative adaptive cruise control (CACC) as their test case and used an automated theorem prover to prove a collision avoidance property under time delay attack on vehicle communication network. Mundhenk et al. [21] used probabilistic model checker - PRISM, for quantifying security vulnerability (in terms of confidentiality, integrity and availability) of automotive architecture at design-time. Before using the model checker, they transformed the automotive communication architecture which consists of ECUs, heterogeneous bus system, and intra-vehicle networks into Continuous-Time Markov Chain with transitions given by exploitability rate and patching rate. However, they did not consider safety of an operating vehicle under adversarial attack. Fawzi et al. [23] and Tiwari et al. [22] proposed detection and prevention methods for an autonomous ground vehicle under sensor spoofing attacks. In [23], secure states were first estimated of a linear control system under sensor spoofing attacks using a decoder. Then, a linear static controller was designed based on the set of secure states. However, this method did not distinguish between noise and attack. Moreover, they assumed that not all sensors can be attacked at the same time. In [22], a learning mechanism was used to construct a set of invariants called “safety envelope”, from collected sensor data. Then, the system was monitored to check if it operated within the safety envelope. In case of a violating, an alarm was raised depending on whether it was an attack or noise. The learning algorithm did not require model of the dynamical system and could distinguish between noise and attack. Unlike the software approaches, Wolf et al. [24] proposed a vehicular hardware security module for protecting

in-vehicle ECUs and their communications.

In this paper, we use networks of priced timed automata (NPTA) with stochastic semantics for modeling an autonomous vehicle under adversarial attack. We consider all sensors of the vehicle are spoofed remotely after determining the initial state of the vehicle. Moreover, we decouple attack from environmental and system noise in our vehicular dynamics model.

IV. Framework for Quantifying Trust in Autonomous System

In this section, we describe our framework for quantifying trust and its various components. At the beginning, we briefly describe the functional software architecture of the autonomous ground vehicle under consideration. Then, we outline the requirements for safe state estimation. Subsequently, we describe the method of our choice for modeling autonomous vehicle under adversarial attack. We verify the controller of the autonomous vehicle against probabilistic safety specification. The confidence interval and confidence percentage obtained from the verification process helps us in quantifying trust on the vehicle.

A. Functional Software Architecture of Autonomous Ground Vehicle

The software architecture of the autonomous ground vehicle (AV) under consideration (See Figure 1) is similar to the ones in [3], [15]. The AV has six on-board sensors, global positioning system (GPS), inertial measurement unit (IMU), cameras, radars, lidars, and ultrasonic. Raw data from these sensors are processed into a form which can be understood by the hierarchical sensor fusion unit. The multi-sensor data fusion algorithm of this unit uses different combination of on-board sensor data for object detection and classification. This data is further combined with information from V2V/V2X communications and on-board maps to enhance perception of the vehicle beyond the capabilities of traditional sensors. The output of the sensor fusion unit is a 3-D map of the environment, which is used by the control and decision unit along with information of the current state of the vehicle to generate an optimal obstacle free trajectory[1]. Subsequently, the trajectory execution sub-system uses propulsion, steering, and braking components to execute the trajectory generated by the control and decision unit. The diagnostic and fault management (D&F) sub-system estimates safe states of the vehicle. It analyzes the generated trajectory to determine whether the vehicle is operating within the safety domain. In case of any deviation from the expected behavior, it issues command to the reactive control sub-system, which immediately responds by taking action such as braking to avoid collision. The reactive control sub-system operates in parallel with the trajectory execution unit and when a threat is identified, its output overrides normal operation of the vehicle.

[1]Note: In Figure 1, the current state refer to a trajectory generated in a previous time step and future state refer to a new trajectory generated by the control and decision unit after analyzing information from sensor fusion unit.

In this paper, we consider an adaptive cruise control unit, whose function is to drive the vehicle in an intended trajectory and avoid collision with other vehicles on the road.

B. Safe State Estimation

We consider an autonomous ground vehicle in which the D&F sub-system monitors the trajectory generated by the control and decision unit to ensure the vehicle operates safely. To perform monitoring, the D&F sub-system estimate a set of safe states based on the knowledge of overall system dynamics and disturbances arising from internal components and external environment.

We first construct an abstract interval around true sensor measurements using *set-membership* method [25]. Then, we estimate the safe states of the system from these sensor values. The abstract interval is constructed by using manufacturers specifications about precision and accuracy of sensors, as well as physical limitations such as sampling jitter and synchronization error. If these information's are not present, then an approximate abstract interval is constructed based on empirical data of initial sensor measurements. Smaller the size of the interval, higher is the confidence on the sensor measurements. Anything outside this abstract interval are bad sensor values (due to adversarial attack) and the trajectory of the system should never enter the bad states (measured from the bad sensor values). Information of safe and bad states enables the D&F sub-system to monitor the system.

C. Modeling of Autonomous Vehicle under Adversarial Attacks

Modeling of systems involving interaction of discrete and continuous dynamics (such as in autonomous ground vehicles) has been successfully carried out using hybrid system [16], [17]. Over the years, several computational frameworks have been developed for representing hybrid system such as hybrid program, hybrid automata (extends the traditional finite state automata by incorporating continuous dynamics), networks of priced timed automata (NPTA), and hybrid petri net. However, these frameworks did not allow randomness in the design. Consequently, the theory of stochastic hybrid system and its modeling formalisms such as Piecewise Deterministic Markov Processes (PMDP), stochastic hybrid automata, and Switched Diffusion Processes (SDP), were developed for capturing variability/uncertainty in the system and performing probabilistic analysis. This enabled modeling and analysis of uncertain autonomous systems.

In this paper, we use NPTA with stochastic semantics [26] for modeling vehicular system dynamics and representing internal/external disturbances including adversarial attack. Using this approach we model the adaptive cruise control (ACC) system of the ground vehicle, which is shown in our case study.

D. Verification of Autonomous Ground Vehicle

When the D&F sub-system of the execution unit (See Figure 1) identifies a suspicious behavior (state), it issues command

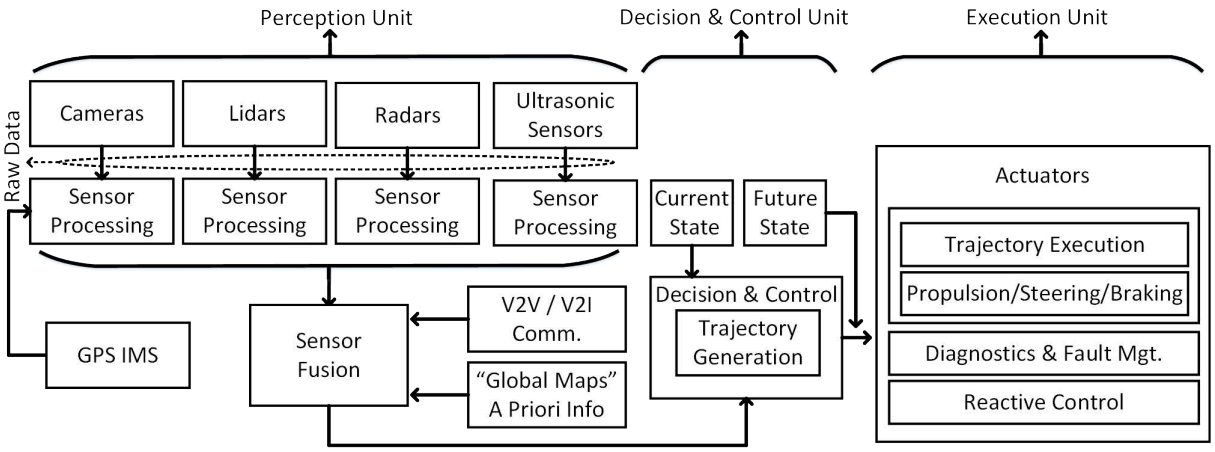


Fig. 1: Functional software architecture of autonomous ground vehicle.

to the reactive control sub-system, which then carries out the actions to prevent fatal scenarios. To ensure the reactive control sub-system behave correctly, we verify that it carries out the safe actions within a probability threshold. As specifications for a controller may be violated once in a while due to internal or external disturbances, we resort to a probabilistic method of verification. With the help of this approach, we can generate probabilistic guarantees on whether the system's state will transition to a safe state within certain time interval.

Model checking is one of the approach for verifying stochsatic hybrid systems [27]. In this approach, a stochastic model $\mathcal{M}$ of a system with an initial state s_0 is expressed as a transition system, behavioral specification such as safety property ϕ is expressed using bounded linear temporal logic, $\theta \in [0,1]$ is a probability threshold, and $\bowtie \in \{>, <, \geq, \leq\}$. Using these notations, the probabilistic model checking problem can be formally stated as $\mathcal{M}, s_0 \models P_{\bowtie\theta}(\phi)$. The underlying algorithm of this technique explores the state-space of the model to decide whether $\mathcal{M}$ satisfies ϕ with a probability $>, <, \geq,$ or $\leq$ to a certain threshold θ. If a case exists where the model does not satisfy the specification, a counterexample in the form of a trace is produced by the model checker.

In our paper, we use Statistical probabilistic Model Checking (SMC) to verify the reactive control sub-system of the autonomous ground vehicle against probabilistic safety specifications (generated from knowledge of system dynamics). The specifications used in this tool are time bounded. Such a model checking approach combines randomized simulation (i.e., Monte Carlo simulation), statistical analysis, and model checking, and it is scalable to large designs. However, using large number of digital clocks (we use one clock) in the system model effects efficiency and scalability of this approach.

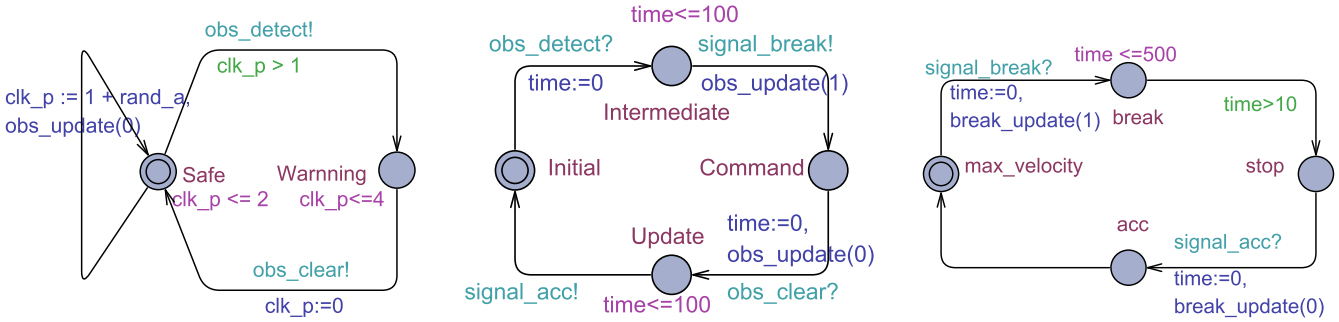
E. Quantification of Trust in the Autonomous Ground Vehicle

Our probabilistic approach for safety verification of the autonomous ground vehicle requires running the system for certain time units to obtain results. Based on the number of runs, we obtain an overall estimate of the correctness of the system. The results are represented in the form of confidence interval $[\theta - \epsilon, \theta + \epsilon]$ (where, $\theta \in [0,1]$ is the probability assigned to the safety specification and ϵ is an approximation parameter) and confidence $(1-\delta)$ (where δ is the confidence parameter). The value of confidence represents the probability of the specification satisfying the system model within the confidence interval for the runs of the system. Higher the value of confidence within the confidence interval, more is the likelihood of the vehicle system model satisfying the safety specification. We write specifications for all the units of the autonomous ground vehicle under consideration and obtain their respective confidence values for a fixed confidence interval. Based on these values, we assign overall trust to the vehicle.

V. Case Study

As a case study, we consider adaptive cruise control (ACC) of an autonomous ground vehicle. ACC relies on data from camera, lidar, or radar to measure distance (d) between a follower car (*fc*) and a leader car (*lc*) on a lane. From distance measurement, velocity and acceleration of the *fc* is calculated. The safety requirement for collision avoidance between *fc* and *lc* is that the distance between them should always be greater than zero i.e. $d > 0$. Under normal operation of the vehicle, the control and decision unit achieves this condition by adjusting the acceleration of *fc*. However, under spoofing attack, the sensors will produce wrong measurements of distance *d*. As a result, the control and decision unit will issue wrong commands to the execution unit, resulting in a possible vehicular collision. To mitigate this issue, diagnostics and fault management sub-system should detect this threat and alert the reactive control sub-system. Subsequently, the later unit should override normal operation and issue the brake command.

In our example, we consider safe distance d between *fc* and *lc* is 20 m. When d is < 20 m between the vehicles, the reactive controller of *fc* should brake within 600 msec to avoid collision. We assume the adversary spoofs the lidar signals, thereby making the sensor detect fake vehicles at a distance



(a) Model of D&F sub-system. (b) Model of reactive controller sub-system. (c) Model of actuator sub-system.

Fig. 2: Model of vehicular sub-system.

$d > 20$m, when it is actually less than the safe distance. Due to this, the reactive controller will not be able to brake within the desired time, resulting in a collision. However, our safe state estimation and monitoring method in the D&F sub-system will detect this fault and notify the reactive controller within the desired time to prevent collision.

To verify this scenario, we use the UPPAL Statistical Model Checking (SMC) tool Version 4.1. We build a very simple timed automata of *D&F sub-system* (Figure 2a) with two states (`safe` and `warning`).The `safe` state indicate that the future state from the control and decision unit conform with the safe state estimated by the D&F sub-system. On the contrary, the `warning` state indicate suspicious behavior. In our current set-up we assume that the adversary cannot modify the D&F sub-system directly, but as the estimation algorithm is not completely accurate, we assign uncertainty to the state variable of the automaton. The `safe` state is initialized with an uncertain variable `clk_p` uniformly between [1.00, 2.00]. The automaton transitions to the `warning` state when the value of `clk_p` is > 1 and it outputs `obs_detected!`, which is sent to the reactive controller sub-system (Figure 2b). From the `warning` state it transition back to the `safe` state when `clk_p<=4` with an output `obs_clear!` and the variable `clk_p` resets to 0. In order to estimate the probability that the `clk_p` will change its value to > 1 within a simulation time bound of 1000 time units, the following specification will be checked.

$$Pr[<= 1000](<> \; clk\_p > 1)$$

The result of the tool for D&F automata is: confidence interval [0.9026, 1.00] with 95% confidence after having generated 36 runs. After the `obs_detect!` signal is received by the *reactive control automata* (Figure. 2b), it transition from `Initial` state to `Command` via `Intermediate` state in < 100 time units. In this transition, the controller sends the command `signal_break` to the actuator sub-system (Figure 2c) as an obstacle has been detected. At the `Initial` state, internal variable `obs` is initialized to 0. After transitioning to `Command` state, the `obs` value changes to 1. This is shown by the method `obs_update(1)` in Figure 2b. To estimate the probability that the internal variable `obs` changes value to 1 in < 100 time units, we use the following specification.

$$Pr[<= 1000](<> \; time < 100 \; and \; obs == 1)$$

The result of the tool for reactive controller automata is: confidence interval [0.9026, 1.00] with 95% confidence after having generated 36 runs. Once the obstacle is clear the automaton transitions to the `Initial` state via `Update` state and the internal variable `obs` value is reset to 0, shown in Figure. 2b as `obs_update(0)`. The output of this transition is the command `signal_acc!`, which is also sent to the actuator sub-system. When the *actuator automata* (Figure. 2c) receives the command `signal_break!` from the reactive control automata, it should transition from `max_velocity` state to the `break` state in < 500 time units. During the same transition, internal variable `apply_break` changes its value from 0 to 1. This is shown by the method `break_update(1)` of Figure. 2c. To estimate the probability that the internal variable `apply_break` changes its value to 1 in < 500 time units, we use the following specification,

$$Pr[<= 1000](<> \; time < 500 \; and \; apply\_break == 1)$$

The result of the tool for actuator automata is: confidence interval [0.9026, 1.00] with 95% confidence after having generated 36 runs. From the `break` state the automata transition to `stop` state in time > 10 time units. During this transition, variable `acc` is initialized to an initial value of 60 with an uncertain derivative `dv` uniformly between [1.00, 2.00]. This uncertainty indicates that the car never comes to a complete halt at the `stop` state. After receiving the command `signal_acc!` from the reactive controller, the vehicle transitions back to the `acc` state. During this transition, the value of internal variable `apply_break` changes from 1 to 0 as shown by the method `break_update(0)` of Figure. 2c. This transition indicates that the collision has been avoided and the vehicle resume its motion. All the safety specifications we considered, were satisfied in the confidence interval [0.9026, 1.00] with 95% confidence after having generated 36 runs.

In Table.1 we have provided summary of trusted and untrusted signals of automatons of Figure. 2a, 2b, 2c. Now, based on the results of these automatons, trust on the autonomous ground vehicle under consideration is 95% for a confidence interval of [0.9026, 1.00].

TABLE I: Summary of Trusted and Untrusted Signals of Automatons

Signals	Untrusted	Trusted	Reason
obs_detect!	✓		Triggered on encountering bad states
obs_clear!		✓	Doesn't depend on bad states
obs_detect?	✓		Relies on obs_detect!
signal_break!		✓	Doesn't depend on bad states
obs_clear?		✓	Doesn't depend on bad states
signal_acc!		✓	Doesn't depend on bad states
signal_break?		✓	Doesn't depend on bad states
signal_acc?		✓	Doesn't depend on bad states

VI. Conclusion

Trust plays an important role in adoption of autonomous systems such as self-driving vehicles. In this paper, we have addressed the trust issue in autonomous systems using an estimation method and statistical model checking approach. The estimation method identifies safe states based on a set-membership method and with the help of system dynamic. The statistical model checking approach provide probabilistic guarantees on whether various sub-systems of the vehicle will satisfy safety specifications within some time interval. In our case study, we use the confidence results generated during verification process to assign trust on the adaptive cruise-control unit of the vehicle. In future, we intend to explain our estimation and verification methods in more sophisticated systems.

Acknowledgment

This work has been partially supported by the National Science Foundation (NSF-1319105) and Army Research Office (W911NF-16-1-0124).

References

[1] DARPA, "Darpa urban challenge," November, 2007.

[2] J. D. Lee and K. A. See, "Trust in automation: Designing for appropriate reliance," *Human Factors*, vol. 46, pp. 50–80, 2004.

[3] S. Behere and M. Torngren, "A functional reference architecture for autonomous driving," *Information & Software Technology*, vol. 73, pp. 136–150, 2016.

[4] O. of the Chief Scientist, "Autonomous horizons: System autonomy in the air force – a path to the future," in *United States Air Force*, 2015.

[5] K. Koscher and et.al., "Experimental security analysis of a modern automobile," in *Proceedings of the 2010 IEEE Symposium on Security and Privacy*, SP '10, pp. 447–462, 2010.

[6] R. Boyle, "Proof-of-concept carshark software hacks car computers, shutting down brakes, engines, and more," in *Popular Science*, May 14, 2010.

[7] S. Checkoway and et. al., "Comprehensive experimental analyses of automotive attack surfaces," in *20th USENIX Security Symposium, San Francisco, CA, USA, August 8-12, 2011, Proceedings*, 2011.

[8] J. Bhatti and T. E. Humphreys, "Covert control of surface vessels via counterfeit civil gps signals," *University of Texas, unpublished*, 2014.

[9] Y. Shoukry, P. Martin, P. Tabuada, and M. Srivastava, "Non-invasive spoofing attacks for anti-lock braking systems," CHES'13, (Berlin, Heidelberg), pp. 55–72, Springer-Verlag, 2013.

[10] J. Petit, B. Stottelaar, M. Feiri, and F. Kargl, "Remote attacks on automated vehicles sensors: Experiments on camera and lidar," in *Black Hat Europe*, November, 2015.

[11] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," in *DEFCON, USA*, August, 2015.

[12] Y. Shoukry, P. Martin, Y. Yona, S. N. Diggavi, and M. B. Srivastava, "Pycra: Physical challenge-response authentication for active sensors under spoofing attacks.," in *ACM CCS*, pp. 1004–1015, ACM, 2015.

[13] N. Navet, Y. Song, F. Simonot-Lion, and C. Wilwert, "Trends in automotive communication systems," *Proceedings of the IEEE*, vol. 93, pp. 1204–1223, June 2005.

[14] P. Hood, "How do google's self-driving cars work?," in *Alphr*, April, 2016.

[15] F. Mujica, "Scalable electronics driving autonomous vehicle technologies," in *Texas Instrument*, April, 2014.

[16] O. Stursberg, A. Fehnker, Z. Han, and B. H. Krogh, "Verification of a cruise control system using counterexample-guided search," *Control Engineering Practice*, vol. 12, no. 10, pp. 1269 – 1278, 2004. Analysis and Design of Hybrid Systems.

[17] S. M. Loos, A. Platzer, and L. Nistor, *FM 2011: Formal Methods: 17th International Symposium on Formal Methods, Limerick, Ireland, June 20-24, 2011. Proceedings*, ch. Adaptive Cruise Control: Hybrid, Distributed, and Now Formally Verified, pp. 42–56. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.

[18] T. Wongpiromsarn, S. Mitra, A. Lamperski, and R. Murray, "Verification of periodically controlled hybrid systems: Application to an autonomous vehicle," *Special Issue of the ACM Transactions on Embedded Computing Systems (TECS)*, vol. 11, no. S2, 2012.

[19] M. Althoff, D. Althoff, D. Wollherr, and M. Buss, "Safety verification of autonomous vehicles for coordinated evasive maneuvers," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*, pp. 1078–1083, June 2010.

[20] B. Zheng, W. Li, P. Deng, L. Gérardy, Q. Zhu, and N. Shankar, "Design and verification for transportation system security," in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1–6, June 2015.

[21] P. Mundhenk, S. Steinhorst, M. Lukasiewycz, S. A. Fahmy, and S. Chakraborty, "Security analysis of automotive architectures using probabilistic model checking," in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1–6, June 2015.

[22] A. Tiwari, B. Dutertre, D. Jovanovic, T. de Candia, P. Lincoln, J. M. Rushby, D. Sadigh, and S. A. Seshia, "Safety envelope for security," in *Proceedings of the 3rd International Conference on High Confidence Networked Systems (HiCoNS)*, pp. 85–94, April 2014.

[23] H. Fawzi, P. Tabuada, and S. Diggavi, "Secure estimation and control for cyber-physical systems under adversarial attacks," *IEEE Transactions on Automatic Control*, vol. 59, pp. 1454–1467, June 2014.

[24] M. Wolf and T. Gendrullis, *Information Security and Cryptology - ICISC 2011*, ch. Design, Implementation, and Evaluation of a Vehicular Hardware Security Module, pp. 302–318. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

[25] M. Milanese and C. Novara, "Set membership identification of nonlinear systems," *Automatica*, vol. 40, no. 6, pp. 957–975, 2004.

[26] A. David, K. G. Larsen, A. Legay, M. Mikučionis, D. B. Poulsen, J. van Vliet, and Z. Wang, *Statistical Model Checking for Networks of Priced Timed Automata*, pp. 80–96. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.

[27] P. Zuliani, C. Baier, and E. M. Clarke, "Rare-event verification for stochastic hybrid systems," in *Proceedings of the 15th ACM International Conference on Hybrid Systems: Computation and Control*, HSCC '12, (New York, NY, USA), pp. 217–226, ACM, 2012.