VCIDS: Collaborative Intrusion Detection of Sensor and Actuator Attacks on Connected Vehicles

Pinyao Guo[1(✉)], Hunmin Kim[2], Le Guan[1], Minghui Zhu[2], and Peng Liu[1]

[1] College of Information Sciences and Technology, Pennsylvania State University, University Park, PA 16802, USA
{pug132,lug14,pliu}@ist.psu.edu
[2] School of Electrical Engineering and Computer Science, Pennsylvania State University, University Park, PA 16802, USA
{huk164,muz16}@psu.edu

Abstract. Modern urban vehicles adopt sensing, communication and computing modules into almost every functioning aspect to assist humans in driving. However, the advanced technologies are inherently vulnerable to attacks, exposing vehicles to severe security risks. In this work, we focus on the detection of sensor and actuator attacks that are capable of actively altering vehicle behavior and directly causing damages to human beings and vehicles. We develop a collaborative intrusion detection system where each vehicle leverages sensing data from its onboard sensors and neighboring vehicles to detect sensor and actuator attacks without a centralized authority. The detection utilizes the unique feature that clean data and contaminated data are correlated through the physical dynamics of the vehicle. We demonstrate the effectiveness of the detection system in a scaled autonomous vehicle testbed by launching attacks through various attack channels.

Keywords: Urban vehicular networks · Intrusion detection
Cyber-physical systems

1 Introduction

Modern urban transportation systems are rapidly evolving toward enhanced intelligence and safety. The evolution has been driven by recent developments in wireless communication, mobile computing, sensing, autonomous driving, etc. In particular, vehicle-to-vehicle (V2V) communications [14] are becoming prevalent in modern vehicles. Real-time traffic information is shared between connected vehicles and provided to drivers such that they can gather better awareness and make more informed decisions to increase traffic safety and efficiency. Recently, major technology companies including Google, Uber and Tesla are leading intensive development of autonomous vehicles [19]. Autonomous vehicles integrate wireless communication, in-vehicle sensing and computing into almost every

X. Lin et al. (Eds.): SecureComm 2017, LNICST 238, pp. 377–396, 2018.
https://doi.org/10.1007/978-3-319-78813-5_19

functioning aspect and provide robust driver-free maneuver in order to handle exhaustive conditions in urban environments.

While the advanced technologies are dedicated to promoting efficiency and safety for vehicles and drivers, they also bring security concerns to the community. Unlike traditional information and communications technology systems such as computers or mobile phones, vehicles are characterized by a strong coupling of the cyberspace where the software runs and the physical world in which they operate. Vulnerabilities rooted from either the cyberspace (e.g., driver backdoor, rootkit) or the physical world (e.g., signal spoofing, wire breakage) could be intentionally exploited by adversaries. Several researchers [15,30,38] conducted jamming, spoofing, and replay attacks on multiple driving guidance sensors including radars, ultrasonic sensors, GPS, etc., on off-the-shelf vehicles. When corrupted sensors are involved in safety-critical decision making, their readings could potentially deceive human drivers or autonomous driving systems and further escalate into disastrous consequences. Furthermore, white-hats recently launched remote hacks into a Jeep Cherokee [21] and multiple models of Tesla [1,2]. The hacks demonstrated the possibility to remotely control vehicular actuators such as steering wheels or gas pedals, which could directly divert vehicles to crashes or severe damages. Given the substantial role security plays in the automotive community, it is imperative to study the attacks before pandemic security problems happen.

In this paper, we focus on the detection of active attacks that are capable of altering vehicle behaviors and directly causing damages to vehicles or drivers. Down to attack consequences, active attacks can be classified into sensor attacks and actuator attacks. *Sensor attacks*, e.g., GPS spoofing, alter authentic sensor readings. *Actuator attacks*, e.g., steering wheel take-over, directly alter control commands to be executed by vehicle wheels. Passive attacks that aim to steal information or break other non-safety aspects are out of our scope.

Table 1. Literature categorization in intrusion detection systems.

Attack sources \ Data sources	Single host	Network
Cyber attacks	host-based IDSs	mobile ad hoc networks wireless sensor networks
Cyber & physical attacks	control-theoretic approaches	*Our solution*

Intrusion detection has been studied extensively in the past decades. Relevant literature can be partitioned into three categories based on their data audit sources and detection capabilities, as shown in Table 1. Traditional host-based IDSs [18,33,36,39] monitor the system behaviors (e.g. filesystem logs, system calls) of a single host. Network-based approaches [7,17,29,32,37,42] from mobile ad hoc networks and wireless sensor networks incorporate networking

traffic in their detection processes. However, both categories are mostly dedicated to the detection of attacks launched within cyberspace. Data corruption attacks launched through physical channels (e.g. sensor spoofing) cannot be detected since no abnormal cyberspace behavior would be triggered and captured. Besides, neither category models the physical mobility of a vehicle, and thus actuator attacks cannot be detected. Control-theoretic approaches [9,12] are proposed to complement existing IDSs. In particular, these approaches leverage the fact that clean and contaminated sensing data and control commands are correlated via physical dynamics of a vehicle. Refer to Sect. 6 for a more comprehensive literature review. A salient limitation of these approaches is that they require one or multiple sensors of a vehicle to be clean. Powerful attackers (as demonstrated in [1,2]) could potentially corrupt all sensors of a vehicle. For instance, an attacker could exploit a backdoor vulnerability in the sensing data processing library and corrupt all sensor readings in a consistent way to avoid the detection.

The goal of this paper is to address the limitation identified in the last paragraph and develop a new IDS for connected vehicles. The key feature of our proposed IDS is the novel integration of V2V communications and control-theoretic approaches. Under the context of connected vehicular networks, V2V communication enables information exchange between nearby vehicles. To our best knowledge, no prior work studies the benefits of V2V communication in intrusion detections of connected vehicles. The paper makes the first attempt to bridge the gap. This paper makes the following contributions:

- We propose a collaborative intrusion detection system, VCIDS, for the detection of sensor and actuator attacks in connected vehicles. The VCIDS fuses local sensing information and that from nearby vehicles to enhance detection capabilities.
- We implement a prototype detection system on a scaled autonomous vehicle testbed and evaluate the system regarding the effectiveness under different attacks launched through multiple attack channels. The results demonstrate detection capabilities under destructive attack cases when all sensors in a vehicle are compromised.

2 Overview

This section presents background information about modern vehicles and vehicle-to-vehicle communication. Then we give an overview of our prior work in intrusion detection for single host and describe its limitation. Finally, we introduce the adversary and defender model considered in the paper.

2.1 Modern Vehicle Platform

A modern vehicle is equipped with a rich set of sensors. In this paper, we only consider the sensors that are related to vehicle motion. Other sensors such as

thermometer or tire pressure monitoring sensor are out of our scope. The sensors related to motion fall into two categories according to their functionalities [34]. *Navigation sensors* such as GPS and inertial measurement units (IMU) serve the purposes of localization and motion tracking. *Observation sensors* such as light detection and ranging sensor (LiDAR) and camera perceive the surroundings of the vehicle and provide information for short-term maneuver such as collision avoidance, lane changing, etc. Observation sensors enable the vehicle to recognize surrounding objects such as nearby vehicles or pedestrians, and measure their relative positions [4]. In order to handle massive data volume and provide real-time control, vehicles are equipped with powerful mobile computing devices (e.g., Nvidia Drive PX 2 [3]) for complicated functionalities such as object recognition. The actuators in a vehicle typically include steering, gas pedal, and brake.

In a vehicle control iteration (shown in Fig. 1), the sensors measure the position and orientation of the vehicle and its surrounding environment. Then the sensor readings are fed to the human driver or the controller inside the vehicle[1]. After that, control commands are generated and executed by the actuators in the physical world. Each step from capturing the physical signals (e.g., electromagnetic waves, acoustic waves) to signal digitization, data processing, and sending the data to the controller/human driver is prone to data corruption. Analogously, the execution of the control commands is also prone to corruption.

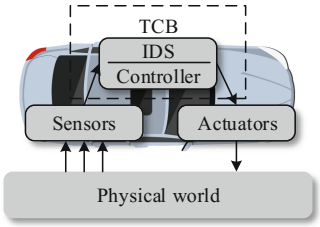


Fig. 1. Vehicle modeling.

2.2 Control Theoretic Approach for Single Host

Modern urban vehicles are cyber-physical systems where the cyberspace (i.e., the computation units) and the physical world in which they operate are strongly coupled. In our prior work [12], we propose a robot intrusion detection system (RIDS) for the detection of sensor and actuator attacks in standalone nonlinear robots. We use a control-theoretic approach and develop a nonlinear unknown input and state estimation (NUISE) algorithm. NUISE exploits the physical dynamics of a single mobile robot and detects attacks by comparing data generated from observed sensor readings and estimates using physical dynamics. In particular, sensor readings can be utilized to estimate new states, and executed control commands can be estimated through state transitions. Therefore, actuator attacks can be detected by comparing planned control commands generated by the controller and executed control commands estimated from sensor readings. With sensor redundancy, sensor attacks can be detected by cross-validating estimated states across the sensors. For the self-containedness of this paper, NUISE algorithm is included in Appendix B.

[1] We do not differentiate controller or human driver in the rest of the paper and refer to them as controller.

In our prior study [12], we evaluate the detection performance of an RIDS implementation against different combinations of sensor and actuator attacks launched from different channels including signal interference, sensor spoofing, logic bomb, etc. The results show that false positive rates and false negative rates are all below 1%, and detection delays are within 0.4 s on average. Other than detection, RIDS also identifies attack types and quantifies attack magnitudes.

The RIDS in [12] only considers local information for the detection of attacks. Hence, it requires that there is at least one clean sensor of each robot. However, this assumption may not be valid for urban vehicles. As demonstrated by [1], attackers can successfully achieve access to a vehicle's Controller Area Network (CAN) and take over multiple functionality modules of a vehicle by sending crafted packages.

2.3 Adversary and Defender Models

We consider adversaries that can launch active attacks that can deviate the vehicles from their normal operation. The adversaries can observe real-time vehicle states and have knowledge about vehicle sensing, actuation, and computing systems. They are capable of launching sensor attacks and/or actuator attacks through different channels, including physical damages (e.g., jamming wheels), signal interference (e.g., GPS spoofing), or cyber breaches (e.g., root-kit) on one or multiple vehicles in a vehicular network. Nevertheless, we assume that for each vehicle in a vehicular network, at least one of its neighboring vehicles has at least one clean observation sensor and a clean localization sensor.

Given the adversary model, the defender has no prior knowledge about the targets of attacks nor the types of attacks. In contrast to previous works [9,12] where at least one clean sensor is required to work, the defender does not trust any particular sensors or actuators nor make assumption on the number of corrupted sensors or actuators on a particular vehicle. As shown in Fig. 1, the controller and the intrusion detection system are treated as a trusted computing base (TCB), which could reside in an isolated computing space such as a separated electric control unit, or be protected with hardware isolation technologies such as TrustZone. We assume each vehicle has clean readings at the very beginning of a trip, and attacks are launched during a trip. The V2V communication channel is assumed to be protected and free of attacks. We do not consider attacks on the communication channel.

2.4 Our Approach: Detection with V2V

V2V is a technology that allows nearby vehicles to exchange assorted information for the safety of urban vehicles and the efficiency of urban traffic. There has been considerable study on applying V2V communication for a variety of functionalities such as collision avoidance [13] and traffic control [8]. A vehicular ad hoc network (VANET) is established to connect nearby vehicles in a decentralized and self-organizing manner. Analogous to mobile ad hoc network, when

a vehicle joins a swarm of vehicles on road, routings are established between the vehicle and other vehicles in the swarm.

To address the limitation in [12], we leverage the information exchange channel as an extra vector for detection. Specifically, with the state estimates and the observations for neighboring vehicles, a vehicle can generate state estimates for nearby vehicles and broadcast the estimates through V2V communication. Leveraging this information, a vehicle can further validate its own state estimates and make salient decisions for itself and other vehicles. Leveraging information from neighbors, the detection could potentially work even when a vehicle has no clean sensor.

3 System Design

The proposed vehicle intrusion detection system follows a distributed and collaborative design, where each connected vehicle in a formed local network participates in intrusion detection without a central authority. The architecture of VCIDS is illustrated in Fig. 2. Each vehicle consists of an IDS node, which is responsible for detecting intrusion locally and collaboratively with nearby vehicles. The VCIDS structure can be divided into several modules. In the IDS node, the monitor collects sensor readings and control commands of the associated vehicle. Utilizing the physical dynamics of the vehicle, the intra-vehicle IDS generates vehicle state estimates and local sensor and actuator attack detection results. Next, the inter-vehicle IDS collects the results from the intra-vehicle IDS module and data transmitted from nearby vehicles to further confirm and detect attacks globally. The global detection also relies on the physical dynamics of the vehicle. In the meanwhile, a secure V2V module transmits and receives information between the vehicles. Upon receiving results from both IDS modules, the decision maker produces conclusive detection results and state estimates for the vehicle controller.

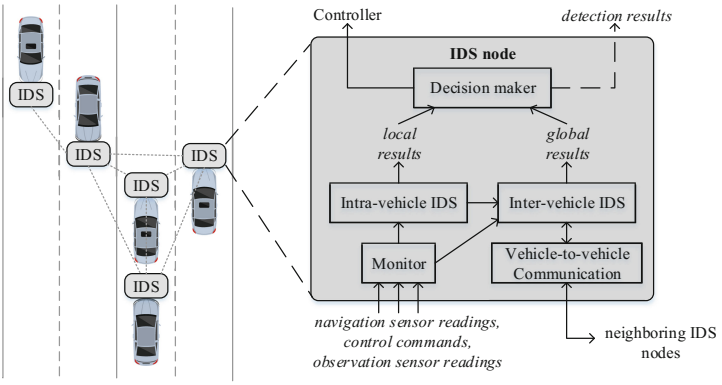


Fig. 2. Vehicle collaborative intrusion detection system architecture.

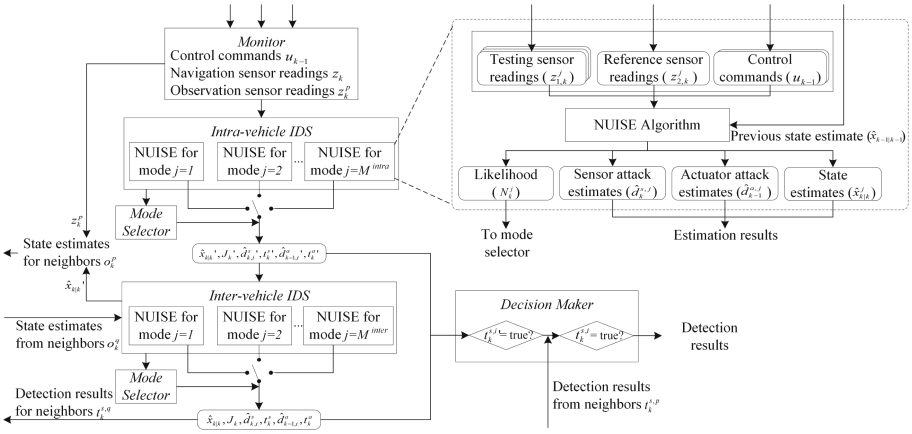


Fig. 3. Vehicle collaborative intrusion detection system schematics.

The VCIDS works iteratively and generates detection results in each control iteration using timely data. Figure 3 shows the schematics of the whole detection system. In the next few subsections, we will first introduce the physical dynamics of the vehicle, and describe how each module works in details.

3.1 Vehicle Physical Dynamics

A vehicle can be modeled as a nonlinear discrete-time dynamic system. Consider current iteration $k-1$, and let $\mathbf{x}_{k-1}$ be the state at the beginning of the current iteration. The controller generates control commands $\mathbf{u}_{k-1}$ and the actuators execute the control commands in the $(k-1)$-th iteration. At the beginning of the k-th iteration, the vehicle reaches the new state $\mathbf{x}_k$ and obtains new sensor readings $\mathbf{z}_k$. Considering sensor and actuator attacks, the system model can be described by the following nonlinear equations:

$$\begin{aligned} \mathbf{x}_k &= f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1} + \mathbf{d}_{k-1}^a) + \zeta_{k-1} \\ \mathbf{z}_k &= h(\mathbf{x}_k) + \mathbf{d}_k^s + \xi_k \end{aligned} \tag{1}$$

where actuator attacks and sensor attacks are modeled as corruptions $\mathbf{d}_{k-1}^a$ and $\mathbf{d}_k^s$, respectively. The first equation in (1) is referred to as the *kinematic model*, which describes vehicle state transitions driven by control command execution. The kinematic model specifies the relation between states and control commands based on the actuator properties, e.g., wheelbase (the distance between the front wheels and the real wheels), engine horsepower, etc. When actuator attacks are launched, the executed control commands deviate from the planned control commands. The deviation is denoted by $\mathbf{d}_{k-1}^a$. The second equation in (1) is the *measurement model*, which describes the relations between sensor readings and vehicle states. The measurement model is determined by the vehicle sensor settings, such as sensors types, sensor placement, etc. When sensor attacks are

launched, the obtained sensor readings deviate from authentic physical values. The deviation is denoted by $\mathbf{d}_k^s$. Vectors ζ_{k-1} are process noises, which account for external disturbances in the kinematic model. Vectors ξ_k are measurement noises, which account for sensing inaccuracy. We assume noise vectors are Gaussian with zero mean and known covariances Q and R, respectively. The kinematic model for a typical rear-wheel-drive vehicle is presented in Fig. 4. The states of the vehicle in a 2D plane include the location and orientation (x, y, θ). The controls include longitudinal velocity and steering (v, ϕ) . The kinematic model for the vehicle can be described as:

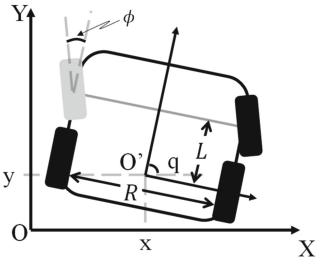


Fig. 4. Kinematic model of a rear-wheel-drive vehicle.

$$
\begin{aligned}
x_k &= x_{k-1} + T(v_{k-1} + d_{k-1}^v)\cos\theta_{k-1} + \zeta_{k-1}^x \\
y_k &= y_{k-1} + T(v_{k-1} + d_{k-1}^v)\sin\theta_{k-1} + \zeta_{k-1}^y \\
\theta_k &= \theta_{k-1} + T\frac{v_{k-1}}{L}\tan(\phi_{k-1} + d_{k-1}^\phi) + \zeta_{k-1}^\theta
\end{aligned}
$$

where $\zeta_{k-1} = [\zeta_{k-1}^x, \zeta_{k-1}^y, \zeta_{k-1}^\theta]^T$ is assumed to be zero mean Gaussian process noise vector, $\mathbf{d}_{k-1}^a = [d_{k-1}^v, d_{k-1}^\phi]^T$ are the actuator attack vectors, L is the wheelbase, and T is the control iteration interval. The sensor measurement model of a vehicle depends on specific sensor types and their configurations. We will introduce the sensor measurement models of our testbed in Sect. 4.2.

3.2 Monitor

In each control iteration, the monitor gathers three types of real-time local data: navigation sensor readings $\mathbf{z}_k$, observation sensor readings $\mathbf{z}_k^p$ ($p \in \mathbb{P}_k$, $\mathbb{P}_k$ denotes the number of nearby vehicles observed at the iteration), and control commands generated from the controller $\mathbf{u}_{k-1}$ (Algorithm 1 line 3–5).

3.3 Intra-vehicle IDS

The intra-vehicle IDS module uses local data to detect sensor and actuator attacks, as well as generate state estimates using local data. In particular, the intra-vehicle IDS applies the *multi-mode estimation algorithm* (Algorithm 2 in Appendix B) on the local data collected from the monitor (line 6).

The multi-mode estimation algorithm maintains a set of possible sensor attack conditions. Each condition is referred to as a *mode*, which represents a hypothesis that a particular sensor is free of attacks, and the remaining sensors are potentially corrupted. The clean sensor is referred to as a *reference sensor*, while the corrupted sensors are referred to as *testing sensors*. Each mode runs a NUISE algorithm (Algorithm 3 in Appendix B) with the corresponding reference sensor readings and testing sensor readings in parallel. The mode set is referred

Algorithm 1. Vehicle Collaborative Intrusion Detection System (VCIDS)

Initialize;
for $k \leftarrow 1$ to ∞ **do**
  Read control commands $\mathbf{u}_{k-1}$;
  Read navigation sensor readings $\mathbf{z}_k$;
  Read observation sensor readings $\mathbf{z}_k^p, p \in \mathbb{P}_k$;
  ▷ **Intra-vehicle IDS**
  Run Algorithm 2 with $(\mathbf{z}_k, \mathbf{u}_{k-1}, \hat{\mathbf{x}}_{k-1|k-1}, \mathbb{M}^{intra})$ and generate $(\hat{\mathbf{x}}'_{k|k}, J'_k, \hat{\mathbf{d}}^{s'}_{k,t}, t^{s'}_k, \hat{\mathbf{d}}^{a'}_{k-1,t}, t^{a'}_k)$;
  ▷ **Inter-vehicle IDS**
  Calculate and broadcast state estimates for neighboring vehicles $\mathbf{o}_k^p$ for $p \in \mathbb{P}_k$ using $\hat{\mathbf{x}}'_{k|k}$ and $\mathbf{z}_k^p$;
  Receive state estimates $\mathbf{o}_k^q$ for $q \in \mathbb{Q}_k$ from neighboring vehicles;
  Run Algorithm 2 with $([(\mathbf{z}_k^{J'_k})^T, (\mathbf{o}_k^q)^T]^T, \mathbf{u}_{k-1}, \hat{\mathbf{x}}_{k-1|k-1}, \mathbb{M}_k^{inter})$ and generate $(\hat{\mathbf{x}}_{k|k}, J_k, \hat{\mathbf{d}}^s_{k,t}, t^s_k, \hat{\mathbf{d}}^a_{k-1,t}, t^a_k)$;
  Broadcast $t_k^{s,q}$ for $q \in \mathbb{Q}_k$;
  Receive $t_k^{s,p}$ for $p \in \mathbb{P}_k$;
  ▷ **Decision maker**
  for each navigation sensor i **do**
    if $t_k^{s,i'} = true$ or $t_k^{s,i} = true$ **then**
      Sensor attack alarm for ith navigation sensor;
    end if
  end for
  if $\exists p \in \mathbb{P}_k$such that$t^{s,p} = true$ **then**
    Sensor attack alarm for observation sensor;
  end if
  if $t_k^a = true$ or $t_k^{a'} = true$ **then**
    Actuator attack alarm;
  end if
  Return $\hat{\mathbf{x}}_{k|k}$ to the controller;
end for

to as $\mathbb{M}$. NUISE generates new vehicle states, attack sizes, and a likelihood for each mode. Detail descriptions of the NUISE algorithm can be found in our prior work [12].

After the NUISE algorithm finishes, a mode selector selects the most probable mode J'_k with the highest likelihood and uses the state estimates $\hat{\mathbf{x}}'_{k|k}$ from the selected mode as the new vehicle state estimates. After that, we further conduct hypothesis testings on the testing sensors and actuators to confirm and identify the attacks $t^{s'}_k$ and $t^{a'}_k$.

3.4 Inter-vehicle IDS

The inter-vehicle IDS is dedicated to confirming the attacks detected by the intra-vehicle IDS, and identifying a boarder range of attacks. The key data source is the observation sensor readings from nearby vehicles. Once the new

state estimates $\hat{\mathbf{x}}'_{k|k}$ is generated, a vehicle can estimate the state of nearby vehicles $\mathbf{o}_k^p$ within the range of its observation sensors (line 7). After that, each vehicle receives the state estimates of itself from nearby vehicles $\mathbf{o}_k^q$ (line 8). Note that the number of observed vehicles $\mathbb{P}_k$ can be different from the number of received state estimates $\mathbb{Q}_k$. Then, the received state estimates $\mathbf{o}_k^q$ are treated as sensor readings from external sources and fed into another round of multi-mode estimation algorithm execution along with the clean sensor readings $\mathbf{z}_k^{J'_k}$ identified by the intra-vehicle IDS (line 9). Finally, the detection results for the received observations $t_k^{s,q}$ are broadcasted, and each vehicle receives the corresponding $t_k^{s,p}$ for decision making, accordingly (line 10–11).

3.5 Decision Maker

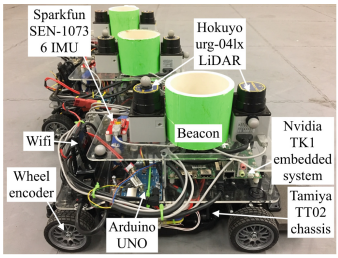
The decision maker confirms attacks using the detection results generated by the intra-vehicle IDS and the inter-vehicle IDS (line 12–23). It checks the detection indexes t generated for navigation sensors, observation sensors, and actuators. Each navigation sensor i is detected to be clean only when both detection indexes $t_k^{s,i'}$ and $t_k^{s,i}$ remain negative. An observation sensor can only be declared as clean when global detection results $t_k^{s,p}$ from all nearby vehicles are not positive. Actuator attacks are positive as long as either $t_k^{a'}$ or t_k^a is positive. Under cases when a vehicle is not connected in a vehicular network, the decision maker still works independently with local detection results. However, the detection results only relying on local data cannot make informed decisions under cases when the observation sensor is corrupted, or all navigation sensors are corrupted with consistent attack vectors.

4 Implementation

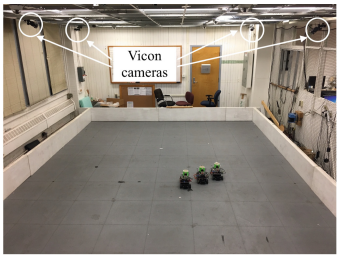
We build a prototype vehicle collaborative intrusion detection system on an indoor testbed which includes three scaled autonomous vehicles. We elaborate on the testbed as follows.

4.1 Testbed Implementation

Figure 5(a) shows the scaled autonomous vehicle testbed on which we implement the VCIDS prototype. Three vehicles are built based on Tamiya TT02 RC car chassis platform [6]. Each vehicle is mounted with a Nvidia Jetson TK1 (Nvidia Cortex-A15) embedded development board [5] as the mobile computing system. TK1 shares the design architecture of vehicular computing systems such as Nvidia PX2, which has been adopted by several autonomous driving manufacturers such as Tesla and Volvo. Its processor features the ARM Architecture and a GPU integration for visual processing intensive applications. The TK1 runs the Robot Operating System (ROS) [31]. Each sensing and actuation module runs in an isolated ROS node (process) and communicates with each other through



(a) Scaled autonomous vehicle testbed with three vehicles.



(b) Indoor experiment environment with Vicon indoor positioning system.

Fig. 5. Scaled autonomous vehicle testbed and indoor positioning system.

sockets. The V2V communication is built on a local wireless ad hoc network. Each vehicle is equipped with a Wifi dongle and joins the ad hoc network after boot. A ROS module is implemented to broadcast or receive observations and detection results generated by the IDSs of other vehicles in the network.

Each vehicle is equipped with four types of sensors: two wheel encoders, a Vicon indoor positioning system (IPS), an IMU (Sparkfun SEN-10736), and two LiDARs (Hokuyo urg-04lx). Two wheel encoders measure the traveling distances of the two rear wheels in a short period of time. They are built with optical sensors fastened on the rear wheels of each vehicle. The optical sensors detect motion of the wheels and communicate with the TK1 through an Arduino board. IPS is powered by Vicon motion capturing system (see Fig. 5(b)), which tracks the position and orientation of each vehicle. An IMU is mounted at the center of each vehicle and provides inertial navigation data. The wheel encoders, IPS and IMU serve as the navigation sensors of an vehicle. Two LiDARs are placed on the top of each vehicle, where one faces the front of the vehicle and the other faces the rear. Each LiDAR scans laser beams in 240 degrees and receives reflection to obtain distances from surrounding objects. The LiDARs together serve as the observation sensors of a vehicle. After processing, they provide a 360-degree distance information of the surroundings of the vehicle.

4.2 Sensor Measurement Models

At each instant of time, navigation sensor readings include data from three sensors: $\mathbf{z}_k = [\mathbf{z}_{k,I}, \mathbf{z}_{k,W}, \mathbf{z}_{k,M}]^T$, where each vector refers to the sensor readings from IPS, wheel encoders, and IMU, respectively. The observation sensors (LiDARs) provides relative distances and directions $\mathbf{z}_k^p$ of nearby vehicles.

Before sensor readings are transmitted to the controller, data go through a processing phase. Navigation sensor readings are converted into vehicle states, i.e., vehicle position (x, y) and heading θ. We convert the raw sensor readings of the wheel encoders and IMU into vehicle states using the measurement models of each sensor. Details of the data processing can be found in Appendix A.

As observation sensors, the LiDARs generate relative position and orientation $\mathbf{z}_k^p$ for each nearby vehicle p. The raw data from a LiDAR includes points that record the ranges of the nearest object from different angles. In the indoor environment surrounded by walls (shown in Fig. 5(b)), we apply the Hough transformation [10] to filter out the range points for the walls (shown as straight lines). After that, remaining range points are clustered and recognized as nearby vehicles. We associate the recognition results with each vehicle using heuristics.

5 Evaluation

In this section, we evaluate the VCIDS on the scaled autonomous vehicle testbed against various attacks and demonstrate its security capabilities. We intend to answer two research questions for the detection system: (1) What benefits does the VCIDS offer in terms of security capabilities? (2) To what extent does the VCIDS influence the detection performance, i.e., effectiveness and efficiency? We compare the detection results generated by intra-vehicle IDS and that by the complete VCIDS.

The three vehicles in the testbed travel in the indoor environment. For the ease of presentation, we label the three vehicles with fixed numbers. In each experiment, vehicle 1 and vehicle 2 circle around the environment in a predefined two-lane road with an identical preset speed of 6 cm/s as shown in Fig. 6(a). Vehicle 3 stays on the roadside without moving, but all onboard sensors are working. During the mission, the three vehicles communicate through V2V communication and collaboratively detect attacks.

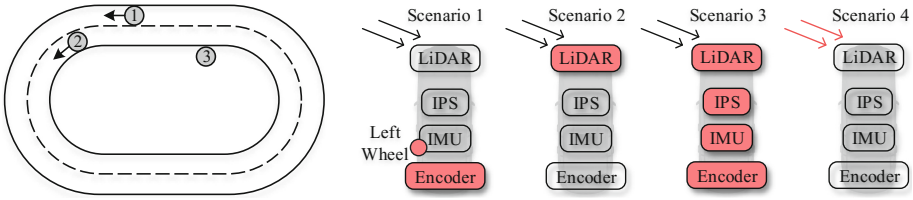
5.1 Attack Scenarios

To demonstrate the effectiveness, we consider the following four attack scenarios where attacks are launched on different targets of the vehicle system. The attack scenarios are conducted independently with each other.

Wheel Encoder Logic Bomb and Wheel Jamming. The attack is launched by replacing the wheel encoder sensor data processing library with a malicious library in vehicle 1. After being triggered at certain instant of time, instead of returning states obtained from motion of the wheel shafts, the malicious library returns the sensor readings with a constant sensor attack vector that shifts the vehicle by -10 cm on the X axis. A plastic stick is placed in the left rear wheel of vehicle 1. The stick adds friction in the wheel and slows down the movement of the wheel (actuator attack).

LiDAR Driver Logic Bomb. Analogous to the wheel encoder sensor logic bomb attack, we add dozens of code in the LiDAR driver program of vehicle 1. After triggering, the customized driver returns fake relative distances and angle measurements of nearby vehicles.

System Hijacking. For advanced attackers, it has been demonstrated that attackers can hijack into the vehicle system and control several components



(a) Scaled autonomous vehicle execution in the indoor environment.

(b) Attack scenarios. Scenario 1: Encoder logic bomb and left wheel jamming. Scenario 2: LiDAR driver logic bomb. Scenario 3: System hijacking. Scenario 4: Rogue nodes.

Fig. 6. Experiment setups.

of a vehicle [1,21]. In order to avoid detection, an attacker would modify all sensor readings in a consistent manner. For instance, an attacker would shift all sensor readings on Y axis by $+10$ cm. During the intra-vehicle detection phase, the multi-mode estimation algorithm does not have a clean sensor as the reference sensor. Moreover, since the sensor readings are corrupted consistently, the hypothesis tests would not raise alarm due to the lack of a clean reference. Here, we launch the attack that corrupts all sensor data in vehicle 1 consistently.

Rogue Nodes. Attackers can setup rogue nodes that broadcast fake messages to nearby vehicles in order to cause wrong decision making for the vehicles. For instance, a rogue node can broadcast a phantom vehicle in front of a vehicle and leads to emergency brakes. In this scenario, we assume that a rogue node is set up by the roadside which broadcasts fake observations. The rogue node broadcasts large amount of fake observations of vehicle 1 that contain shifted observations.

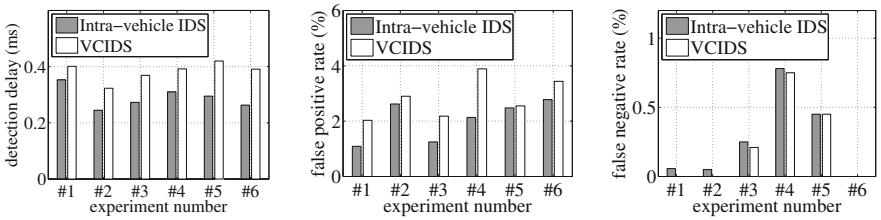
5.2 Detection Results

In order to demonstrate the security capabilities of the VCIDS, we compare the detection results generated by the intra-vehicle IDS (i.e., a standalone single host IDS that does not leverage information from neighboring vehicles) and the complete VCIDS. Table 2 shows the detection results against the four attack scenarios we launch in the testbed. We observe that the intra-vehicle IDS can only detect the first attack scenario when a subset of navigation sensors are under attacks. On the contrary, the VCIDS detects all attack scenarios. When the observation sensors are under attacks (Scenario 2), state estimates for nearby vehicles are corrupted. When vehicle 2 and vehicle 3 receive corrupted observations from vehicle 1, their inter-vehicle IDSs raise sensor attack alarms and send the results $t_k^{s,2}$ and $t_k^{s,3}$ back to vehicle 1. When all sensor readings in vehicle 1 are corrupted consistently (Scenario 3), the intra-vehicle IDS of vehicle 1 does not raise any alarm. However, the observations from vehicle 2 (o_k^3) and vehicle 3 (o_k^3) used in the inter-vehicle IDS of vehicle 1 are inconsistent during inter-vehicle IDS execution. Under rogue nodes attack (scenario 4) when fake nodes

broadcast erroneous observations, the inter-vehicle IDS raises alarms due to the inconsistencies between observations from other vehicles and the results from intra-vehicle IDS[2].

Table 2. Detection results from intra-vehicle IDS and VCIDS.

Attack scenario	Attack type (channel)	Detected by intra-vehicle IDS	Detected by VCIDS
Wheel encoder logic bomb+wheel jamming	Sensor+actuator (cyber+physical)	Yes	Yes
LiDAR driver logic bomb	Sensor (cyber)	No	Yes
System hijacking	Sensor (cyber)	No	Yes
Rogue nodes	Sensor (cyber)	No	Yes



(a) Detection delay. (b) False positive rate. (c) False negative rate.

Fig. 7. Detection performance comparison between results from intra-vehicle IDS and VCIDS against wheel encoder logic bomb & wheel jamming attack.

To investigate the detection performance in terms of detection delay and accuracy, we launch attacks that can be detected by the intra-vehicle IDS and compare the results. A false positive refers to an instant of time that an alarm is raised for a clean sensor, and a false negative refers to an instant of time that alarm is not raised for a corrupted sensor. Figure 7 shows the comparison for detection delays, false positive rates and false negative rates. We notice that detection delays for VCIDS are larger since the VCIDS requires more steps after the intra-vehicle detection. We also notice a slight increases on the false positive rates and a decrease on the false negative rates. All rates are below 4%.

Sensor noises determine the accuracies of state estimates, attack estimates, and decision making. In our approach, sensor noises are modeled with unbound support and propagate along with each calculation step in Algorithms 2 and 3.

[2] A detailed explanation on why the NUISE algorithm can determine which mode reflects the authentic values is provided in [12] Sect. 5.2.

The detection system makes decisions under certain level of confidence. Therefore, a vehicle equipped with more accurate sensors could identify attacks with better performance.

6 Related Works

Intrusion Detection for CPSs. Control theory has been utilized to detect sensor attacks for linear cyber-physical systems in recent works [9,22–24]. Several works [11,27,28,40] study both actuator and sensor attacks for linear cyber-physical systems. In contrast, most real world vehicles are modeled as nonlinear systems. In [11,23,24,27], processing and measurement noises rooted in actuators and sensors are not considered or considered with bounded support. In contrast, real world vehicles are subject to stochastic noises with unbounded support. Guo et al. [12] propose NUISE that handles sensor and actuator attack for nonlinear system with unbounded support. However, it requires at least one clean sensor on a single host. Some works study attack-detection on networked systems [20,22,25,26,41]. However, these studies either share the limitations in previously mentioned single host-based solutions or use voting mechanisms in detection making. For instance, Park et al. [25] use Kalman filter to obtain estimates of local agent and use t-tester to leverage inter-observations between neighboring agents to statistically validate estimates. The approach is restricted to linear systems under sensor attacks without actuator attacks. Moreover, the t-testers work under the assumption that majority of the agents are attack-free.

Attacks Targeted on Vehicles. Yan et al. [38] successfully conduct jamming and spoofing attacks on the driving guidance sensors including radars, ultrasonic sensors and forward-looking cameras on a Tesla Model Petit et al. [30] present effective jamming, replay, relay, and spoofing attacks on camera and LiDAR sensors. It has been demonstrated that civilian GPS are vulnerable to spoofing or jamming [15,35]. Several groups demonstrate the possibility to remotely control multiple subsystems in latest off-the-shelf vehicle models such as Tesla [1,2] and Jeep Cherokee [21].

7 Conclusion

The advanced technologies applied in modern vehicles bring both opportunities and security concerns for the community. In this study, we propose a vehicle collaborative intrusion detection system (VCIDS) for the detection of sensor and actuator attacks which target connected vehicles. The detection leverages the physical dynamics of vehicles and utilizes the correlation between clean and contaminated data to estimate attacks. We build a prototype system on a scaled autonomous vehicle testbed and test the system against several types of attacks launched through different attack channels. The results demonstrate that VCIDS can achieve better security capabilities over a single host IDS. Leveraging information from neighboring vehicles, VCIDS works under destructive attack cases

even when all the sensors of a vehicle are compromised. VCIDS can promote the resilience of vehicles against attacks. We plan to investigate intrusion response strategies for urban vehicles as our future works.

Acknowledgement. This work was supported by NSF CNS-1505664, ARO W911NF-13-1-0421 (MURI) and ARO W911NF-15-1-0576.

Appendix A Data Processing with Measurement Models

IPS. The IPS sensor directly measures and returns the states of a vehicle.

Wheel encoder. The raw data measured by the wheel encoders are the distances traveled by each wheel (l_L, l_R). In data processing phase, we convert them into vehicle states using previous states $\mathbf{x}_{k-1}$: $x_k = x_{k-1} + (l_L + l_R)\cos\theta_k/2$, $y_k = y_{k-1} + (l_L + l_R)\sin\theta_k/2$, $\theta_k = \theta_{k-1} + (l_R - l_L)/R$, where R is the distance between the left and the right wheel.

IMU. The IMU sensor generates a quaternion $[q_0, q_1, q_2, q_3]^T$, a 3-D acceleration $\mathbf{a}_{k,M}^{local}$, and a 3-D rotational speed $\mathbf{w}_{k,M}^{local}$ on body-fixed coordinate. We first obtain coordinate transformation matrix from body-fixed coordinate to global coordinate [16]:

$$C(q) = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}.$$

Acceleration vector and rotation speed on the global coordinate system can be obtained as $C(q)\mathbf{a}_{k,M}^{local}$ and $C(q)\mathbf{w}_{k,M}^{local}$, respectively. Vehicle velocity vector can be updated by: $\mathbf{v}_k = [v_{k,M}^x, v_{k,M}^y, v_{k,M}^z]^T = \mathbf{v}_{k-1} + \mathbf{a}_k^{global}T$. Then the state vector can be calculated by integration as follows: $x_k = x_{k-1} + v_{k,M}^x T + \frac{1}{2}a_{k,M}^x T^2$, $y_k = y_{k-1} + v_{k,M}^y T + \frac{1}{2}a_{k,M}^y T^2$, $\theta_k = \theta_{k-1} + w_{k,M}^z T$.

After the data processing phase for each sensor, sensor readings transmitted to the controller are in the form of vehicle states. For navigation sensors, we have: $\mathbf{z}_{k,i} = \mathbf{x}_k + \mathbf{d}_{k,i}^s + \xi_{k,i}, i = I, W, M$, where $\mathbf{d}_{k,i}^s = [d_{k,i}^{s,x}, d_{k,i}^{s,y}, d_{k,i}^{s,\theta}]^T$, $\xi_{k,i} = [\xi_{k,i}^x, \xi_{k,i}^y, \xi_{k,i}^\theta]^T$ refer to attack vectors and measurement noises for each navigation sensor, respectively.

Appendix B Algorithms

Algorithms 2 and 3[3] are proposed in the Appendix of [12]. We include them here to be self-contained.

[3] Notations † and $|\cdot|_+$ refer pseudoinverse and pseudodeterminant, respectively.

Algorithm 2. Multi-mode Estimation Algorithm

Input: Sensor readings $\mathbf{z}_k$; control commands $\mathbf{u}_{k-1}$ from control module; previous state estimates $\hat{\mathbf{x}}_{k-1|k-1}$; mode set $\mathbb{M}$

Output: State estimate; attack vector estimates; mode estimate; confirmed attack indices t_k^s, and t_k^a;

Set parameters $w_s, w_a, c_s, c_a, \alpha_s, \alpha_a$;
Initialize;
for mode $j \in \mathbb{M}$ **do**
    Run NUISE (Algorithm 3) with input ($\mathbf{u}_{k-1}$, $\hat{\mathbf{x}}_{k-1|k-1}$, $\mathbf{z}_{1,k}^j$, $\mathbf{z}_{2,k}^j$, P_{k-1}^x) and generate ($\hat{\mathbf{x}}_{k|k}^j$, $\hat{\mathbf{d}}_k^{s,j}$, $\hat{\mathbf{d}}_{k-1}^{a,j}$, $P_k^{x,j}$, $P_k^{s,j}$, $P_{k-1}^{a,j}$, $\mathcal{N}_k^j$);
    $\bar{\mu}_k^j \leftarrow \max\{\mathcal{N}_k^j \mu_{k-1}^j, \epsilon\}$;
end for
for mode $j \in \mathbb{M}$ **do**
    $\mu_k^j \leftarrow \frac{\bar{\mu}_k^j}{\sum_{i=1}^{|\mathbb{M}|} \bar{\mu}_k^i}$;
end for
Sensor mode selection $J_k \leftarrow \text{argmax}_j \mu_k^j$;
Obtain estimates and covariance matrices from J_k: $\hat{\mathbf{x}}_{k|k} \leftarrow \hat{\mathbf{x}}_{k|k}^{J_k}$, $\hat{\mathbf{d}}_k^s \leftarrow \hat{\mathbf{d}}_k^{s,J_k}$, $\hat{\mathbf{d}}_{k-1}^a \leftarrow \hat{\mathbf{d}}_{k-1}^{a,J_k}$, $P_k^x \leftarrow P_k^{x,J_k}$;
$b_k^s \leftarrow (\hat{\mathbf{d}}_k^{s\,T}(P_k^{s,J_k})^{-1}\hat{\mathbf{d}}_k^s > \chi^2_{p=|\hat{\mathbf{d}}_k^s|}(\alpha_s))$;
$b_k^a \leftarrow (\hat{\mathbf{d}}_{k-1}^{a\,T}(P_{k-1}^{a,J_k})^{-1}\hat{\mathbf{d}}_{k-1}^a > \chi^2_{p=|\hat{\mathbf{d}}_{k-1}^a|}(\alpha_a))$;
if $b_k^s = True$ **and** $\sum_{i=0}^{w_s-1} b_{k-i}^s \geq c_s$ **then**
    for each testing sensor t in mode J_k **do**
        Sensor attack vector estimate for testing sensor t: $\hat{\mathbf{d}}_{k,t}^s = \sum_{i=0}^{w_s-1} \hat{\mathbf{d}}_{k-i,t}^s / w_s$;
        if $\hat{\mathbf{d}}_{k,t}^{s\,T}(P_{k,t}^{s,J_k})^{-1}\hat{\mathbf{d}}_{k,t}^s \geq \chi^2_{p=|\hat{\mathbf{d}}_{k,t}^s|}$ **then**
            $t_k^s = 1$; confirm sensor attack on sensor t;
        end if
    end for
end if
if $b_k^a = True$ **and** $\sum_{i=0}^{w_a-1} b_{k-i}^a \geq c_a$ **then**
    $t_k^a = 1$; confirm actuator attack;
end if
    return $\hat{\mathbf{x}}_{k|k}$; based on the confirmations of attacks, find a new mode J_k; sensor attack vector estimates $\hat{\mathbf{d}}_{k,t}^s$ with t_k^s ($t \in$ {testing sensors in mode J_k}); actuator attack vector estimates with t_k^a $\hat{\mathbf{d}}_{k-1,t}^a$ ($t \in \{1, \cdots, n\}$);

Algorithm 3. Nonlinear Unknown Input and State Estimation Algorithm

Input: $\mathbf{u}_{k-1}$, $\hat{\mathbf{x}}_{k-1|k-1}$, $\mathbf{z}^j_{1,k}$, $\mathbf{z}^j_{2,k}$, P^x_{k-1}

Output: $\hat{\mathbf{x}}^j_{k|k}$, $\hat{\mathbf{d}}^{s,j}_k$, $\hat{\mathbf{d}}^{a,j}_{k-1}$, $P^{x,j}_k$, $P^{s,j}_k$, $P^{a,j}_{k-1}$ $\mathcal{N}^j_k$

Initialize;

▷ **Actuator attack vector $\mathbf{d}^{a,j}_{k-1}$ estimation**

$\tilde{P}^j_{k-1} \leftarrow A^j_{k-1} P^x_{k-1} (A^j_{k-1})^T + Q^j_{k-1}$;

$\tilde{R}^{*,j}_{2,k} \leftarrow C^j_{2,k} \tilde{P}^j_{k-1} (C^j_{2,k})^T + R^j_{2,k}$;

$M^j_{2,k} \leftarrow ((G^j_{k-1})^T (C^j_{2,k})^T (\tilde{R}^{*,j}_{2,k})^{-1} C^j_{2,k} G^j_{k-1})^{-1} \ (G^j_{k-1})^T (C^j_{2,k})^T (\tilde{R}^{*,j}_{2,k})^{-1}$;

$\hat{\mathbf{d}}^{a,j}_{k-1} \leftarrow M^j_{2,k} (\mathbf{z}^j_{2,k} - C^j_{2,k} f(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}))$;

$P^{a,j}_{k-1} \leftarrow M^j_{2,k} \tilde{R}^{*,j}_{2,k} (M^j_{2,k})^T$;

▷ **State prediction**

$\hat{\mathbf{x}}^j_{k|k-1} \leftarrow f(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1} + \hat{\mathbf{d}}^{a,j}_{k-1})$;

$\bar{A}^j_{k-1} \leftarrow (I - G^j_{k-1} M^j_{2,k} C^j_{2,k}) A^j_{k-1}$;

$\bar{Q}^j_{k-1} \leftarrow (I - G^j_{k-1} M^j_{2,k} C^j_{2,k}) Q^j_{k-1} (I - G^j_{k-1} M^j_{2,k} C^j_{2,k})^T + G^j_{k-1} M^j_{2,k} R^j_{2,k} (M^j_{2,k})^T (G^j_{k-1})^T$;

$P^{x,j}_{k|k-1} \leftarrow \bar{A}^j_{k-1} P^x_{k-1} (\bar{A}^j_{k-1})^T + \bar{Q}^j_{k-1}$;

▷ **State estimation**

$\tilde{R}^j_{2,k} \leftarrow C^j_{2,k} P^{x,j}_{k|k-1} (C^j_{2,k})^T + R^j_{2,k} + C^j_{2,k} G^j_{k-1} M^j_{2,k} R^j_{2,k} + R^j_{2,k} (M^j_{2,k})^T (G^j_{k-1})^T (C^j_{2,k})^T$;

$L^j_k \leftarrow (C^j_{2,k} P^{x,j}_{k|k-1} + R^j_{2,k} (M^j_{2,k})^T (G^j_{k-1})^T)^T (\tilde{R}^j_{2,k})^{-1}$;

$\hat{\mathbf{x}}^j_{k|k} \leftarrow \hat{\mathbf{x}}^j_{k|k-1} + L^j_k (\mathbf{z}^j_{2,k} - h^j_2(\hat{\mathbf{x}}^j_{k|k-1}))$;

$P^{x,j}_k \leftarrow (I - L^j_k C^j_{2,k}) P^{x,j}_{k|k-1} (I - L^j_k C^j_{2,k})^T + L^j_k R^j_{2,k} (L^j_k)^T - (I - L^j_k C^j_{2,k}) G^j_{k-1} M^j_{2,k} R^j_{2,k} (L^j_k)^T - L^j_k R^j_{2,k} (M^j_{2,k})^T (G^j_{k-1})^T (I - L^j_k C^j_{2,k})^T$;

▷ **Sensor attack vector $\mathbf{d}^{s,j}_k$ estimation**

$\hat{\mathbf{d}}^{s,j}_k \leftarrow \mathbf{z}^j_{1,k} - h^j_1(\hat{\mathbf{x}}^j_{k|k})$;

$P^{s,j}_k \leftarrow C^j_{1,k} P^{x,j}_k (C^j_{1,k})^T + R^j_{1,k}$;

▷ **Likelihood of the mode**

$\nu^j_k \leftarrow \mathbf{z}^j_{2,k} - h^j_2(\hat{\mathbf{x}}^j_{k|k-1})$;

$\bar{P}^j_{k|k-1} \leftarrow C^j_{2,k} P^{x,j}_{k|k-1} (C^j_{2,k})^T + R^j_{2,k} - C^j_{2,k} G^j_{k-1} M^j_{2,k} R^j_{2,k} - R^j_{2,k} (M^j_{2,k})^T (G^j_{k-1})^T (C^j_{2,k})^T$;

$n^j \leftarrow rank(\bar{P}^j_{k|k-1})$;

$\mathcal{N}^j_k \leftarrow \frac{1}{(2\pi)^{n^j/2} |\bar{P}^j_{k|k-1}|^{1/2}_+} \exp(-\frac{(\nu^j_k)^T (\bar{P}^j_{k|k-1})^\dagger \nu^j_k}{2})$;

References

1. Car Hacking Research: Remote attack Tesla motors. Keen Security Lab of Tencent (2016). http://keenlab.tencent.com/en/2016/09/19/Keen-Security-Lab-of-Tencent-Car-Hacking-Research-Remote-Attack-to-Tesla-Cars/
2. Chinese group hacks a Tesla for the second year in a row (2017). https://www.usatoday.com/story/tech/2017/07/28/chinese-group-hacks-tesla-second-year-row/518430001/

3. Introducing the new NVIDIA DRIVE PX 2 for autocruise driving and HD mapping (2017). http://www.nvidia.com/object/drive-px.html
4. Nvidia driveworks software development kit for self driving cars (2017). https://developer.nvidia.com/driveworks
5. NVIDIA Jetson TK1 (2017). http://www.nvidia.com/object/jetson-tk1-embedded-dev-kit.html
6. RC TT02 Chassis - TT02 Factory Finished (2017). https://www.tamiyausa.com/items/radio-control-kits-30/rc-semi-assembled-chassis-35900/rc-tt02-chassis-57984?product-id=57984
7. Axelsson, S.: The base-rate fallacy and the difficulty of intrusion detection. ACM Trans. Inf. Syst. Secur. (TISSEC) **3**, 186–205 (2000)
8. Bauza, R., Gozalvez, J., Sanchez-Soriano, J.: Road traffic congestion detection through cooperative vehicle-to-vehicle communications. In: 2010 IEEE 35th Conference on Local Computer Networks (LCN) (2010)
9. Bezzo, N., Weimer, J., Pajic, M., Sokolsky, O., Pappas, G.J., Lee, I.: Attack resilient state estimation for autonomous robotic systems. In: IROS (2014)
10. Duda, R.O., Hart, P.E.: Use of the Hough transformation to detect lines and curves in pictures. Commun. ACM **15**, 11–15 (1972)
11. Fawzi, H., Tabuada, P., Diggavi, S.: Secure estimation and control for cyber-physical systems under adversarial attacks. IEEE Trans. Autom. Control **59**, 1454–1467 (2014)
12. Guo, P., Kim, H., Virani, N., Xu, J., Zhu, M., Liu, P.: Exploiting physical dynamics to detect actuator and sensor attacks in mobile robots. arXiv preprint arXiv:1708.01834 (2017)
13. Hafner, M.R., Cunningham, D., Caminiti, L., Del Vecchio, D.: Cooperative collision avoidance at intersections: algorithms and experiments. IEEE Trans. Intell. Transp. Syst. **14**, 1162–1175 (2013)
14. Hartenstein, H., Laberteaux, L.: A tutorial survey on vehicular ad hoc networks. IEEE Commun. Mag. **46**(6) (2008)
15. Humphreys, T.E., Ledvina, B.M., Psiaki, M.L., OHanlon, B.W., Kintner Jr., P.M.: Assessing the spoofing threat: development of a portable GPS civilian spoofer. In: Proceedings of the ION GNSS International Technical Meeting of the Satellite Division (2008)
16. Kuipers, J.B., et al.: Quaternions and Rotation Sequences. Princeton University Press, Princeton (1999)
17. Lee, W., Stolfo, S.J., et al.: Data mining approaches for intrusion detection. In: USENIX Security (1998)
18. Lichodzijewski, P., Zincir-Heywood, A.N., Heywood, M.I.: Host-based intrusion detection using self-organizing maps. In: Proceedings of the 2002 International Joint Conference on Neural Networks, IJCNN 2002 (2002)
19. Litman, T.: Autonomous vehicle implementation predictions. Implications for transport planning (2014). http://www.vtpi.org/avip.pdf
20. Manandhar, K., Cao, X., Hu, F., Liu, Y.: Detection of faults and attacks including false data injection attack in smart grid using Kalman filter. IEEE Trans. Control Netw. Syst. **1**, 370–379 (2014)
21. Miller, C., Valasek, C.: Remote exploitation of an unaltered passenger vehicle. Black Hat, USA (2015)
22. Mo, Y., Garone, E., Casavola, A., Sinopoli, B.: False data injection attacks against state estimation in wireless sensor networks. In: 2010 49th IEEE Conference on Decision and Control (CDC) (2010)

23. Pajic, M., Tabuada, P., Lee, I., Pappas, G.J.: Attack-resilient state estimation in the presence of noise. In: 2015 54th IEEE Conference on Decision and Control (CDC) (2015)
24. Park, J., Ivanov, R., Weimer, J., Pajic, M., Lee, I.: Sensor attack detection in the presence of transient faults. In: ICCPS (2015)
25. Park, P., Khadilkar, H., Balakrishnan, H., Tomlin, C.J.: High confidence networked control for next generation air transportation systems. IEEE Trans. Autom. Control **59**, 3357–3372 (2014)
26. Pasqualetti, F., Carli, R., Bullo, F.: Distributed estimation via iterative projections with application to power network monitoring. Automatica **48**, 747–758 (2012)
27. Pasqualetti, F., Dorfler, F., Bullo, F.: Attack detection and identification in cyber-physical systems. IEEE Trans. Autom. Control **58**, 2715–2729 (2013)
28. Patel, A., Alhussian, H., Pedersen, J.M., Bounabat, B., Júnior, J.C., Katsikas, S.: A nifty collaborative intrusion detection and prevention architecture for smart grid ecosystems. Comput. Secur. **64**, 92–109 (2017)
29. Paxson, V.: Bro: a system for detecting network intruders in real-time. Comput. Netw. **31**, 2435–2463 (1999)
30. Petit, J., Stottelaar, B., Feiri, M., Kargl, F.: Remote attacks on automated vehicles sensors: experiments on camera and LiDAR. Black Hat, Europe (2015)
31. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y.: ROS: an open-source robot operating system. In: ICRA workshop on open source software (2009)
32. Roesch, M., et al.: Snort: lightweight intrusion detection for networks. In: LISA (1999)
33. Ryan, J., Lin, M.J., Miikkulainen, R.: Intrusion detection with neural networks. In: Advances in Neural Information Processing Systems (1998)
34. Schweber, B.: The autonomous car: a diverse array of sensors drives navigation, driving, and performance (2017). http://www.mouser.com/applications/autonomous-car-sensors-drive-performance/
35. Volpe, J.: Vulnerability assessment of the transportation infrastructure relying on the global positioning system (2001)
36. Warrender, C., Forrest, S., Pearlmutter, B.: Detecting intrusions using system calls: alternative data models. In: Proceedings of the 1999 IEEE Symposium on Security and Privacy (1999)
37. Wu, Y.S., Foo, B., Mei, Y., Bagchi, S.: Collaborative intrusion detection system (CIDS): a framework for accurate and efficient IDS. In: 2003 Proceedings of 19th Annual Computer Security Applications Conference (2003)
38. Yan, C., Xu, W., Liu, J.: Can you trust autonomous vehicles: contactless attacks against sensors of self-driving vehicle. In: 24th DEFCON Hacking Conference (2016)
39. Yeung, D.Y., Ding, Y.: Host-based intrusion detection using dynamic and static behavioral models. Pattern Recognit. **36**, 229–243 (2003)
40. Yong, S., Zhu, M., Frazzoli, E.: Resilient state estimation against switching attacks on stochastic cyber-physical systems. In: CDC (2015)
41. Zeng, W., Chow, M.Y.: Resilient distributed control in the presence of misbehaving agents in networked control systems. IEEE Trans. Cybern. **44**, 2038–2049 (2014)
42. Zhang, Y., Lee, W.: Intrusion detection in wireless ad-hoc networks. In: Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (2000)