2022 IEEE Intelligent Vehicles Symposium (IV)
June 5-9, 2022. Aachen, Germany

CVGuard: Mitigating Application Attacks on Connected Vehicles

Ahmed Abdo[1], Guoyuan Wu[2], Qi Zhu[3],Nael Abu-Ghazaleh[1]

Abstract— **Connected vehicle (CV) applications promise to revolutionize our transportation systems, improving safety and traffic capacity while reducing environmental footprint. Many CV applications have been proposed towards these goals, with the US Department of Transportation (USDOT) recently initiating some designated deployment sites to enable experimentation and validation. While the focus of this initial development effort is on demonstrating the functionality of a range of proposed applications, recent attacks have demonstrated their vulnerability to *application level attacks*. In these attacks, a malicious actor operates within the application's parameters but providing falsified information. This paper explores a framework that protects against such application-level attacks. Then, we analyze the impact of the attacks, showing that an individual attacker can have substantial effects on the safety and efficiency of traffic flow even in the presence of message security standards developed by USDOT, motivating the need for our defense. Our defense relies on physically modeling the vehicles and their interaction using dynamic models and state estimation filters as well as reinforcement learning. It combines these observations with knowledge of application rules and guidelines to capture logic deviations. We demonstrate that the resultant defense, called CVGuard, can accurately and promptly detect attacks, with low false positive rates over a range of attack scenarios for different CV applications.**

Keywords: **Connected Vehicles, Cyber-security, State estimation, Security credential management system**

I. Introduction

By leveraging the wireless communications, connected vehicles (CVs)[1] can optimize their operations to act as a coordinated ensemble rather than individual vehicles. They may communicate directly with each other (vehicle-to-vehicle, or V2V), with roadside infrastructure (vehicle-to-infrastructure, or V2I), and with the "Cloud" (V2C). The US Department of Transportation (USDOT)[2] has identified potential benefits of CV applications, including improved safety, mobility, system efficiency, and reduced environmental footprints such as energy consumption, mobile source emissions and noises. Many CV applications start being prototyped and have reference implementations[3]. The USDOT has been testing applications in three deployment sites[4]. The focus of developers has been primarily on performance, and cyber-security is not being considered deeply. CVs expose a large attack surface as an open system with many participants and complex functionalities: attacks may target application protocols, networking, sensing, and vehicle control, potentially resulting in accidents, traffic delays, and other harm to the system. A public key certificate management standard, the Secure Certificate Management System (SCMS) [5], has been defined by USDOT. However, SCMS only ensures the eligibility of those certified cars and roadside units that may participate in communications [6], and it is expected that these certificates would be available widely, thus easily obtained by attackers. As a result, it does not guarantee that CV applications are risk-free since an attacker with a certificate can participate in the system and send malicious messages (e.g., with falsified information). General security mechanisms to protect CV systems have been proposed to harden CV ecosystem including leveraging cryptography[7][8], signature-based detection[9], and malware detection[10].

Some defenses have been proposed for application level attacks. Still, they suffer from several limitations, such as being specific to a single application, and having overheads incompatible with real-time deployment.

This paper develops the first defense against application level attacks which can be customized to different applications under realistic assumptions and practical overheads. In addition, the proposed defense leverages information from two domains: in-vehicle and inter-vehicle, to address the threat model. The first component of the proposed defense estimates the physical dynamics of the vehicle to provide a starting point for reasoning about safety within a physical context. Physics-based models have been proposed as a defense against sensor tampering attacks[11]: a physics-based model of the vehicle is used to predict its state and to detect anomalous sensor readings as a deviation between measurements and predictions. However, since our threat model is substantially different (a remote attacker can falsify information about a state that is not local to the vehicle and therefore beyond the reach of its physics based model), we use this estimate in a different way in combination with other techniques to detect consistency within the region of operation.

Since the threats relate to other vehicles, the defense also captures inter-vehicle anomalies. Specifically, we use Reinforcement Learning (RL) as an inter-CV security framework that allows the system to learn the CV behavior to ensure that vehicles do not collide with each other. Finally, we use application specific logic detectors to detect protocol interactions inconsistent with the allowed maneuvers and policies defined in the application.

Moreover, the implementation is lightweight since it does

[1]Ahmed Abdo and Nael Abu-Ghazaleh are with the Department of Electrical and Computer Engineering, University of California, Riverside, CA 92521, USA. aabdo003@ucr.edu, nael@cs.ucr.edu.
[2]Guoyuan Wu is with the Center for Environmental Research and Technology, University of California, Riverside, CA 92521, USA. gywu@cert.ucr.edu .
[3]Qi Zhu is with the Department of Electrical and Computer Engineering, Northwestern University, Evanston, IL 60201, USA. qzhu@northwestern.edu.

not require keeping a substantial state or history, making the defense suitable for real-time application. Our results show that the proposed defense can: a) mitigate different cyber-physical attacks; (b) detect the attacks promptly; and (c) have a low false positive rate. As a result, we believe that our defense can significantly improve the security of CV systems against both known and emerging application-level attacks.

We evaluate the defense on various application scenarios for multiple CV applications. We show that it can successfully prevent published attacks with low overheads.

In summary, the contributions of this paper include:

- We introduce CVGuard, a new defense against application-level attacks on CV systems. The defense models both vehicle level dynamics and inter-vehicle interactions to enable the detection of application-level attacks.
- We model a number of application-level attacks and use them to evaluate CVGuard, showing its effectiveness in preventing these attacks.
- We perform both simulation-based and real-world evaluation of CVGuard.

II. Threat Model

We consider a CV environment where attackers can compromise a device that is eligible to obtain a certificate from SCMS and participate in the system. SCMS prevents an attacker from spoofing other CVs' identifiers. However, our concern is with those attacks where malicious actors participate in CV protocol by replaying valid messages or sending them with fabricated data. In this study, we do not consider attacks that target message delays, jamming, physical attacks on sensors or controllers. We also do not consider mitigation against attacks that exploit bugs in the software stack of any existing components running on the infrastructure or vehicles.

We assume that vehicles communicate through Dedicated Short Range Communication (DSRC) devices based on IEEE 802.11p[12]. However, the proposed system does not limited to DSRC. Equipped vehicles with onboard units (OBUs) can send Basic Safety Messages (BSMs) to other equipped vehicles or infrastructure consisting of two parts: BSMs-Part 1 are transmitted periodically (typically every 100 msec), containing critical data elements such as vehicle size, position, speed, heading, acceleration, brake system status. BSMs-Part 2 are event-based messages, containing various optional data elements customized by different manufacturers, e.g., ABS activated.

On the other hand, Road Side Units (RSUs)are used to coordinate behaviors across cars or to maintain certain shared states.

We consider the following types of attacks.

- **Fake Message Attack** where the adversary starts to create messages' fields or parameters such as velocity, position, and acceleration, with injected biased values, and then broadcasts the messages to harm the entire system.
- **Replay Message Attack** where the adversary receives and stores a beacon that is broadcast by the other vehicle, and replays it at a later time with malicious intent. The replayed beacon contains old information that can lead to hazardous effects.
- **Stealthy Message Attack** where the adversary is aware of the vehicle dynamics and hence employs its parameters to avoid normal detection filters that can easily find random malicious behaviors. To create the position field in this attack, we use the following equation:

$$lp_{t+1} = lp_t + s_t * \Delta_t + a_t * \Delta_t^2/2$$

where lp, s, a, t, and $t+1$ are the lane position, speed, acceleration, recent time step, and next time step, respectively. This equation calculates position using speed, acceleration, and the time difference between the recent and previous times. The vehicle's speed increases with a_t at each time step until it reaches v_{max}. The vehicle's speed cannot exceed the speed limit of the roadway segment.

III. CV Applications

In this section, we highlight some of the CV applications that are used as case studies in this paper.

In **Intelligent Traffic Signals** (I-SIG), RSUs host I-SIG to manage intersections adaptively and intelligently with goals such as reducing collision, decreasing idle time, and improving traffic flows. The BSM messages are broadcast by CVs and captured by a trajectory awareness process that sustains the latest trajectory for each vehicle linked to its vehicle ID. The signal planning process monitors the traffic signal status. Then, it develops a signal plan based on the incoming real-time trajectory data that are fed into the COP (Controlled Optimization of Phases) algorithm[13][14]. After planning, the I-SIG controller sends signal control commands to the controller. The COP algorithm estimates each vehicle's arrival time and uses dynamic programming to calculate the optimal signal plan with the least (estimated) total delay.

Cooperative Ramp Merging System: In Cooperative Ramp Merging, the V2I system localizes vehicles on a virtual map to be used later by different control algorithms to provide driver assistance to enable safe and smooth merging. The output of the merging control algorithm contains the recommended speed for any cooperative vehicles in the merging process. In particular, the reference accelerations are calculated by a feedforward/feedback control algorithm [15] and sent to the vehicles, so they can regulate the gaps smoothly even before reaching the merging point. As a result, rear-end or sideswipe collisions in the conflict zone are essentially eliminated. Finally, **Cooperative Adaptive Cruise Control** (CACC) utilizes V2V communication to form platoons of vehicles that travel with closer spacing, reducing aerodynamic drag, and improving roadway capacity[16], [17], [18]. In this implementation, the Platoon Management Protocol (PMP) [19] controls platoon operations and maneuvers. The leading vehicle acts as the coordinator and controls platoon

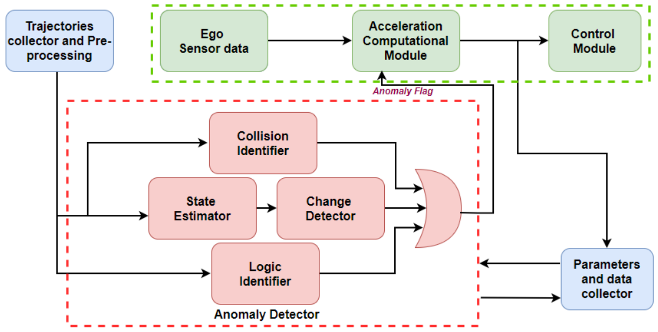
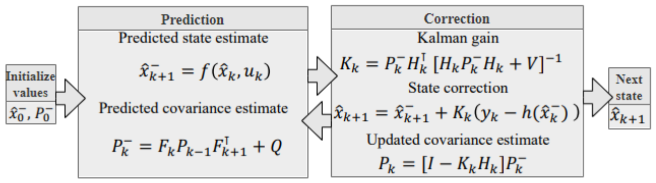


Fig. 1: An example of CVGuard architecture used in CACC. It consists of the blocks within the red dotted line boundaries used for . The onboard architecture for CACC consists of the blocks within the green dotted line boundaries. Acceleration Computation Module computes desired acceleration. The actuarial control module computes braking pressure and motor torque.



Initialize values
$\hat{x}_0^-, P_0^-$
Prediction
Predicted state estimate
$\hat{x}_{k+1}^- = f(\hat{x}_k, u_k)$
Predicted covariance estimate
$P_k^- = F_k P_{k-1} F_{k+1}^T + Q$
Correction
Kalman gain
$K_k = P_k^- H_k^T [H_k P_k^- H_k + V]^{-1}$
State correction
$\hat{x}_{k+1} = \hat{x}_{k+1}^- + K_k(y_k - h(\hat{x}_k^-))$
Updated covariance estimate
$P_k = [I - K_k H_k] P_k^-$
Next state
$\hat{x}_{k+1}$

Fig. 2: The state estimation algorithm

maneuvers such as speed, lane change, and merging with other platoons.

IV. CVGuard Architecture

Application-level attacks rely on communicating falsified information to manipulate a car or a group of cars into an attacker's desired action. The essential question to defend against such attacks is how to determine whether or not the information received from remote CVs is trustworthy. We break down this problem into three smaller problems: (1) Is the predicted state of the car consistent with the measured state? (2) Does acting on the received information potentially lead to collisions or other dangerous actions? And (3) Are the received application-level actions consistent with the application's logic (e.g., do they reflect valid maneuvers?). The components of our defense try to answer these questions, respectively, to detect application-level attacks from different perspectives: when they deviate from the application logic; when they cause the measured and predicted states to diverge; and when they lead to dangerous situations such as collisions. Figure 1 gives an onboard architecture of our proposed CVGuard system. It consists of: (1) **State Estimator**; (2) **Change Detector**; (3) **Collision Identifier**; and (4) **Logic Identifier**.

In the *State Estimator* algorithm, external attacks can be potentially detected by checking whether the (perceived) physical states of the vehicle are consistent with its expected states determined by its dynamics and control. It is defined by the dynamic system properties and control algorithm, mathematically represented by control invariants. The Change Detector can detect unusual changes in states based on a cumulative sum of recursive residual statistics. It allows the system to detect attacks accurately and rapidly, avoiding false positives due to transient errors. Even though individual messages seem to be acceptable by the *State Estimator* algorithm, they can still lead to collisions. This could be caused by stealthy attackers who launch attacks that maximize the damage to the system without being detected to cause a crash or cause other safety problems. The *Collision Identifier* predicts the location of nearby vehicles to detect potential collisions or other hazards. Finally, the Logic Identifier ensures that a protocol or maneuver output does not compromise the safety even under attacks, which relates to under-specified or incomplete protocol logic. For example, if an application logic fails to consider corner cases, such as a large gap between two platoons before merging, CVGuard considers it a V2V anomaly. The logic identifier is specific to each application.

A. State Estimator

An accurate state estimator that tracks vehicles with changing dynamics can be achieved by using multiple filter models that provide estimates of some variables such as velocity and acceleration. Our state estimator adopts an *Interacting Multiple Model* (IMM) [20] that estimates updated states and state covariances based on the ensemble of the most common models (e.g., constant velocity, constant acceleration, constant turn, etc.). Every model is a single Kalman filter that re-initializes with mixed state estimates and covariance based on their probabilities of "switching to" or "mixing with" each other. Thus, constantly correcting each filter to reduce its residual error, even when it does not represent the true motion of the object. In this way, an IMM filter can switch to an individual model based on the specific vehicle dynamics without waiting for convergence first. Thus, using these models can adequately predict the tracked vehicle's possible motions, which is better than using only one model over time.

The different models of the IMM, such as constant velocity or constant acceleration models, follow the same steps of the extended Kalman filter. However, they differentiate in using the dynamical equations in the "predict state".

In general,the algorithm [21] is divided into two main procedures: prediction and correction. The first component takes the last estimation $\hat{x}_k$ and the current input u_k, and generates a prediction $\hat{x}_{k+1}$. However, this prediction is refined using the received data. Similarly, the covariance matrix of the estimation error P_k^- (i.e., the error between the real states x_k and the estimated states $\hat{x}_k$) is predicted using the process covariance matrix Q and the state transition matrix F_k. The second procedure uses the previous predictions $\hat{x}_k^-$, P_k^-, the observation matrix H_k, and the covariance of the measurement noise V, to compute the Kalman gain K_k, which is defined as the uncertainty in a predicted state divided by the uncertainty in predicted state plus uncertainty in measurement readings or messages. Therefore, the state prediction is corrected using the measurement and the covariance matrix is updated. The output of this procedure $\hat{x}_{k+1}$ and P_k will feed the next iteration of the algorithm as $\hat{x}_k^-$ and P_k^-.

The inputs of the EKF can be the position and velocity in a time step. At the same time, the outputs can be the estimated position and velocity that the vehicle may happen in the next time step. The goal of using the EKF is to combine any instantaneous reading (which could be maliciously injected values) with the observed dynamics of the system, allowing us to identify unreasonable/inconsistent measurements rapidly. The EKF structure is shown in Fig. 2.

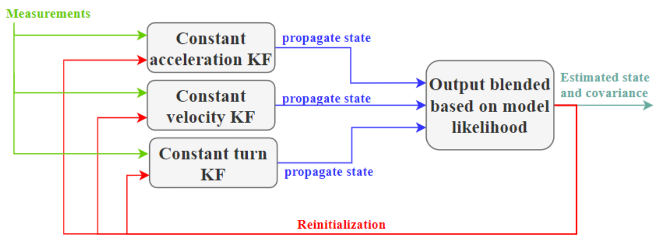


Fig. 3: Overview of the State Estimator

B. Change Detector

The change detector is essential to detect errors from compromised (or noisy) sensors by characterizing the signals changes and controlling the overall error rate. Thus, we propose using the Cumulative Sum of Recursive Residual (CUSUM) [22] change detector that continuously monitors the error of the regression model. A significant increase in the error is interpreted as a change in the distribution that generates the examples over time. When a change is detected, the actual regression model is deleted, and a new one is constructed. The CUSUM statistic is described by the following recurrent equation:

$$S_i(k+1) = S_i(k) + |r_i(k)| - b_i \tag{1}$$

Where r_i is the prediction residual associated with each sensor and S_i is the anomaly score. $S_i(0) = 0$ and $b_i > 0$ are selected to prevent $S_i(k)$ from increasing when there are no attacks. When $S_i(t_k) > \tau_i$, an alarm associated with sensor i is triggered where τ_i is the threshold value.

C. Collision Identifier

The state estimator is a lightweight solution that efficiently detects cyber-physical attacks on an individual CV. However, we need to identify and avoid potential collisions and other undesirable conditions. Thus, we use Reinforcement Learning (RL) [23] to learn the proper maneuver sequence that can help detect anomalies even if the dynamics of each connected vehicle appear to be accurate. The *Collision Identifier* RL algorithm is based on three components: state, action, and reward. The state describes the current condition of an agent. The action is what the agent can do in each time step. Finally, the reward describes positive or negative feedback from the environment due to the agent's action. The overall goal of RL is to learn a policy that maximizes the total reward of an agent through learning from the states and actions when it interacts with the environment. In our *Collision Identifier* algorithm, we use Q-learning [24], [25] because of its combination of effectiveness and simplicity. Q-learning is a value-based learning algorithm that updates its value function based on the Bellman principle. First, we create a Q-table where the agent can update its item by learning the rewards associated with all state-action pairs, based on the following equation at each time step:

$$Q^{new}(s_t, a_t) = (1-\alpha).Q(s_t, a_t) + \alpha \times (r_t - \gamma.max_a Q(s_{t+1}, a))$$

where α is the learning rate (ranging from 0 to 1) and represents how much the agent should learn from a new observation; r_t is the acquired reward for any taken action; γ is the discount factor that controls how much each reward can affect our decision; $max_a Q(s_{t+1}, a))$ is the estimated reward from the next action where the agent selects the optimal action to maximize the reward; $Q(s_t, a_t)$-values are the estimated values, and they represent how much the agent expects to get after performing an action. s is the current state of the vehicle. Since the goal of RL is to maximize the long-term rewards through maneuvers, we design the state, action, and reward as follows:

State: We design our states as a set of possible situations where the vehicle can inhabit during movement. These states represent also neighbor vehicles around the vehicle of interest (also called ego vehicle) during an action, and its nearby detected vehicles are called remote vehicles. In our model, we consider a total possible number of the ego vehicle's states or s_t of 16 discretizing the different configurations of nearby vehicles. For example, s_1 represents an ego vehicle that has only one vehicle in front of it.

Action: We define the action space $A = \{action_1, action_2\}$ to refer to the two possible action patterns of an agent in the system, reflecting either normal or abnormal behavior. The action classification is based on **Time-to-Collision (TTC)** while observing the CV traffic system; if TTC deviates outside preset threshold reflecting normal operation, then we consider the system to be anomalous (i.e., $action_2$); otherwise, we consider it normal ($action_1$). The action is used in the Q-table to define the probability that the agent takes a normal behavior or not. **TTC** is a metric [26] that measures the time taken for a vehicle to collide with the vehicle in front of it, which is an important metric to measure how safe CV components are under cyber attacks quantitatively. TTC of vehicle i at instant t can be calculated as follows,

$$TTC_i(t) = \frac{D_i(t) - D_{i-1}(t) - l_i}{V_i(t) - V_{i-1}(t)}$$

where $D_i(t)$ and $V_i(t)$ stand for the position and speed of vehicle i, respectively, at instant t, and l_i is the length of the vehicle i.

Reward: We design our reward scheme to have a minimal positive value for safe actions and a large negative value if any safety violation occurs. The reward value is determined by TTC of the agent or ego vehicle, as shown below. The $Threshold$ value is estimated throughout experiments to be 0.5 seconds.

$$r_t = \begin{cases} 1, & \text{if } TTC > Threshold. \\ -10, & \text{otherwise.} \end{cases} \quad (2)$$

The RL algorithm is shown in Algorithm 1.

Algorithm 1 Reinforcement learning training process

```
1: procedure UPDATE $Q$ AND CALCULATE LOSS FUNCTION
2:     Initialize $Q$ and $S$ values
3:     while $S_t$ is not terminated do
4:         if $A_t$ violates the reward policy then
5:             $R_t$ = big negative reward
6:         else
7:             $R_t$ = small positive reward
```

D. Logic Identifier

To ensure that a CV protocol or maneuver does not compromise the safety of the included CVs under attack, we have to practice a plausibility check functionality or safety policies. To achieve this, it is necessary to start with a systematic study of the main properties of each CV maneuver since the discovery of such characteristics can most generally affect the security of their corresponding implementation instances. In Algorithm 2, we develop a simplified protocol in which the model updates the timer according to the event. This event represents all the known CV events or maneuvers. Every event in $EventRange_i$ triggers the same update on $timer[i]$. The $Retrieve$ function reacts with the CV environment to pick different properties if such a property is available. These properties assure the CV protocol's safety and include space gaps, relative locations, relative velocities, lane consistency, etc. For example, in the reference CACC implementation, an anomaly can be generated if the space gap does not reach the value below the maximum safe threshold. Thus, ensuring the robustness and safety of the protocol algorithm under the different application-level attacks.

Algorithm 2 A simplified Logic Identifier Algorithm

```
1: EventRange = $(0 \cdots (N-1))$
2: TimerIndexRange = $(0 \cdots (M-1))$
3: Property = $(0 \cdots (P-1))$
4: while $number\ of\ CVs \neq 0$ do
5:     for $\forall\ event \in EventRange$ do
6:         timer = $[i \in TimerIndexRange \mapsto None]$
7:         if $timer[i] > 0$ then
8:             $timer[i] - 1$                          ▷ count down
9:             $RETRIEVE(Property)$
10:        else
11:            TIMEOUT                                 ▷ expire
```

V. Evaluation

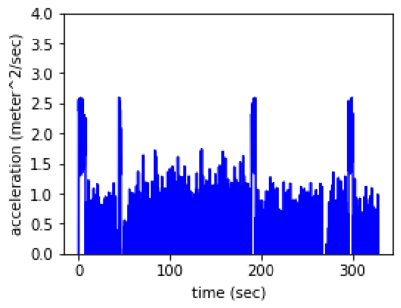
A. Experimental Setup

To evaluate CVGuard, we used VENTOS (VEhicular NeTwork OpenSimulator) [27], which is an extension of Veins simulation [28]. VENTOS enables us to design, test, and evaluate different traffic scenarios. It integrates a C++ simulator for studying vehicular traffic flows and combines two widely used simulators, Simulation of Urban Mobility (SUMO)[29] and OMNET++[30]. SUMO is an open-source road traffic simulator that uses Traffic Control Interface (TraCI) to communicate simulation commands.

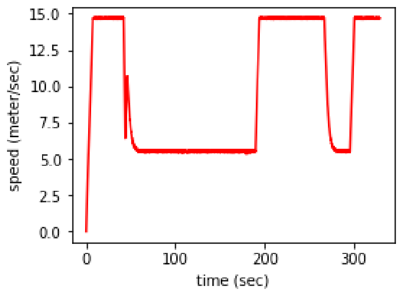
B. CVGuard Effectiveness

One of the most challenging aspects of using simulator data is fidelity. Unfortunately, no repository of sufficient real-world driving data from various driving scenarios is available. To address this issue, we monitored the vehicles data over time coming from the road traffic simulator (SUMO) and compared it with the HighD dataset [31] that provides trajectory data corresponding to actual vehicles driving on German highways. We validated that the SUMO data distribution is consistent with HighD with several metrics. We carried out the following experiments to demonstrate how we built and configured the state estimation component of CVGuard. Different vehicular dynamics characteristics can be used to create the IMM algorithm. However, using a large number of models impacts performance negatively. Thus, we wanted to build an IMM algorithm with the best-suited and most efficient configurations under different trajectory segments while driving over time to improve the tracking accuracy and model switching speed of maneuvering target tracking. In the simulation, we tested different vehicles and measured metrics such as velocities and steering behaviors, as shown in Figures 4a, 4b, and 4c. These three selected dynamics characteristics show that the vehicle switches mostly its dynamics between constant velocity, constant acceleration, and turning right or left to reach its destination. As a result, three Kalman filter models have been selected to create and test the IMM algorithm based on the analysis of the dynamics characteristics evaluation. These models are a constant velocity (CV), a constant acceleration (CA), and a three-dimensional turn with a kinematic constraint (TURN) Kalman filters. Adding additional models to IMM can lead to overfitting problems and degrade detection performance.

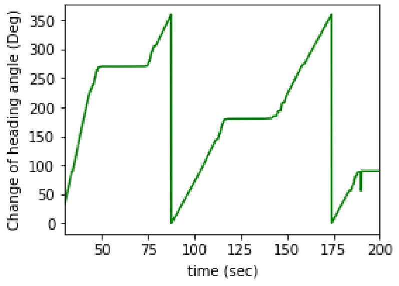
To evaluate this phase, we made the malicious vehicle manipulate its parameters and broadcast its forged or fake messages to the nearest connected vehicles. These BSM messages can contain false synthetic variables such as velocity and acceleration. Thus, the state estimator filter produces noticeably significant prediction errors. Therefore, it can accurately predict the surrounding connected vehicles' states, as shown in Fig. 7. To detect the presence of the cyber-attacks and filter out the transient errors caused by physical disturbances (e.g., winds), we use a change detector component that is based on the non-parametric cumulative sum (CUSUM) anomaly detector to uncover the false data



(a) A constant acceleration (CA) example



(b) A constant velocity (CV) example



(c) Turning with a kinematic constraint (TURN) example

Fig. 4: Tracking velocity, acceleration and change of heading angle for a vehicle while testing.

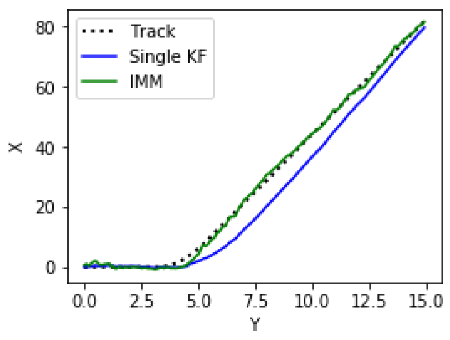


Fig. 5: Improved tracking performance by IMM tracker (with three dynamic models – CV, CA, kinematic constraint (TURN)) compared to single model based tracker

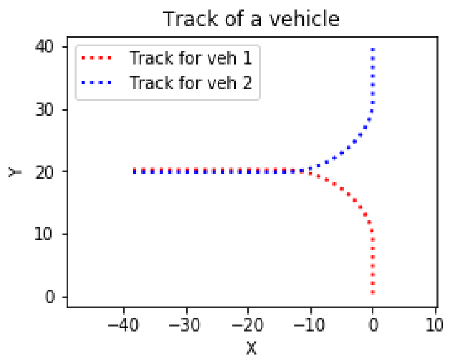
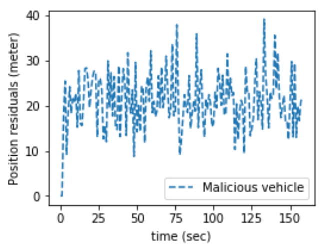


Fig. 6: Two vehicles entering same lane at intersection. Vehicles start on the right hand side of the figure and follow the track towards the left hand side.

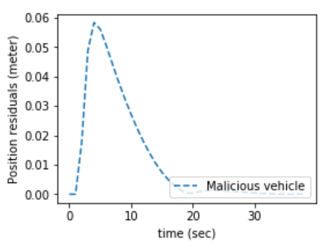
injection attacks, as shown in Fig. 8. From Fig. 5, we can see that tracking a connected vehicle based on the IMM filter is highly maneuverable while predicting the future location of the connected vehicle, and it outperforms the single filter model. The single constant acceleration filter model has the more significant position and rate errors, and it is slow to recover non-maneuvering error levels after the maneuver ends. Even though the IMM tracker in the state estimator phase has superior performance in detecting trajectories anomalies than a single model tracker, it cannot catch all kinds of attacks. For example, two vehicles in an intersection can enter the same lane or road and collide if they have replayed trajectories messages or produced stealthy attack messages, thus fooling the IMM tracker since the RMS errors will be minor. Therefore, the collision identifier phase uses Time-to-Collision(TTC) metric as an action that can assist in detecting such scenarios, as shown in Fig. 6.

To build an efficient collision identifier, we generated all possible scenarios that an ego vehicle can interact with other remote CVs, which are defined as different RL states as described in Section IV. These RL states were utilized in the learning process to update and finalize the reinforcement learning Q-table in the collision identifier so that CVGuard uses collision identifier at each time step to detect *Replay* or *Stealthy* attacks. *Stealthy* attack is challenging to be detected since it is based on the physics of the vehicular system, and it is used to maximize the damage to the system while avoiding detection. Thus, we consider the worst-case scenario for CVGuard in which an attacker is undetected while injecting falsified information into the system continuously. For example, Fig. 9 shows that the next state estimator filter failed to detect any malicious activity due to the normal values of the measured position prediction errors. However, the collision identifier was able to detect this attack through observing the low values of ”Q-value” of the RL algorithm, which represent the action outputs of a connected vehicle's states during this attack, as shown in Fig. 10.

Moreover, we tested CVGuard against application-level attacks presented in [32]. These attacks include: 1) merging over large distances attack where the attacker is located between two platoons such that it can communicate with both platoons simultaneously and deceive ranging sensor by pretending that it is a member of the front platoon;



(a) Measured position residuals for a malicious vehicle.



(b) Measured position residuals for another malicious vehicle.

Fig. 7: Different position residuals cases that are measured by the state estimator component.

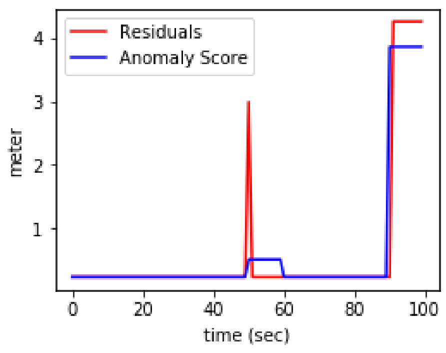


Fig. 8: The detection statistic of the change detector component in CVGuard.

2) merging across different lanes attack where a malicious vehicle is in front of the rear platoon and sends messages pretending to be a part of the other platoon (in another lane); 3) platoon takeover attack where an attacker transmits fake messages of a fake platoon so that the rear platoon merges with the attacker. Thus, this platoon becomes under the attackers' control and can be manipulated dangerously. As shown in Fig. 11, the space gap values are too large during the attack so that the logic identifier component in CVGuard is able to detect such attacks.

Finally, we conducted a series of experiments to compare our CVGuard accuracy with the other anomaly detector such as in [33] in terms of false positive and true positive rates. In general, the accuracy of the classifier is critical to make a security-based decision since the longer we wait, the less valuable an anomaly alert will be. The anomaly detector in [33] targets CVs' security only in CACC applications, and it is based on machine learning. Fig. 12 shows that the Receiver Operating Characteristic (ROC) curve for our CVGuard detection system outperforms the other scheme by 3%. In particular, CVGuard can detect attacks with a probability close to 1 while having a low false alarm rate (less than 3%). In practice, we can lower the false positive rate by requiring multiple anomalous detection before raising an alarm, although this could delay detection.

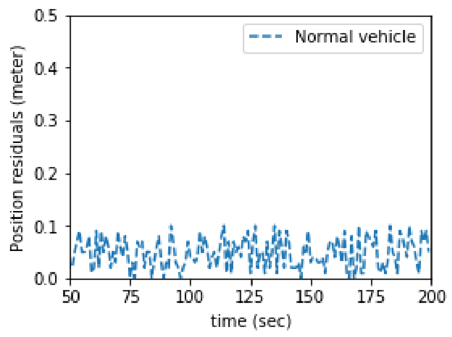


Fig. 9: Measured position residuals (stealthy attack).

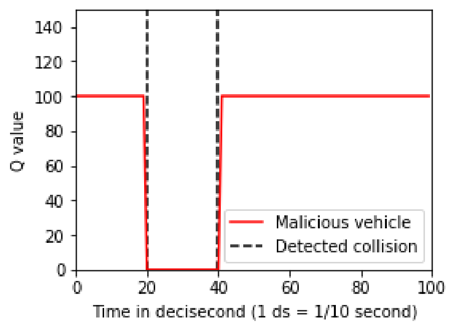
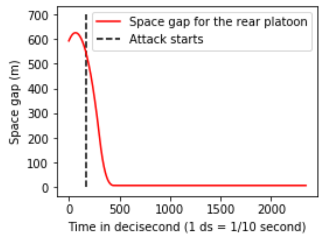


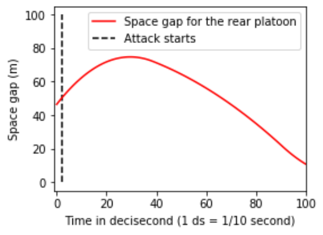
Fig. 10: The Q-value of the reinforcement learning algorithm used in the collision detector component.

VI. Concluding Remarks

Connected Vehicles (CVs) are an emerging field in transportation that brings new opportunities to improve safety, mobility/efficiency, and sustainability. However, many applications are still not considering their security vulnerabilities. The attacks that exploit the vulnerabilities of these communication protocols may lead to a complete reversal of the benefits promised by CVs. Toward this end, the deployment of CV applications has a long way to go before they are reliably safe from attacks. This paper presented a new comprehensive framework for detecting different attacks against CVs, based on state estimation, maneuvers monitoring, and reinforcement learning techniques. Our evaluation of the mitigation framework showed that our mitigation can mitigate the introduced attacks, making it a promising approach



(a) Gap between two platoons under Merge over distances attack.



(b) Gap between two platoons under Platoon takeover.

Fig. 11: Examples of attacks against CVs applications.

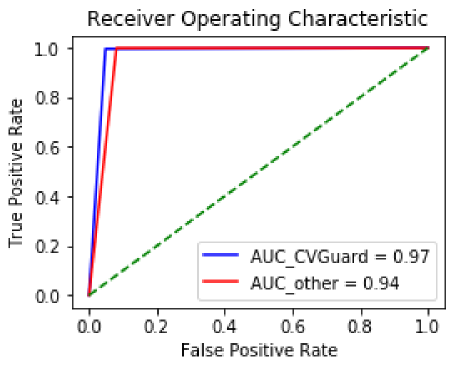


Fig. 12: ROC curve for detection strategy.

to support the system resilience of CV applications. We will extend our defense framework to multi-agent connected drones in the future. Moreover, we will investigate designing a secure recovery mechanism or technique to maintain the targeted vehicles against such attacks.

ACKNOWLEDGEMENTS

This material is partially supported by the National Science Foundation (NSF) grant IIS-1724341. It is also partially supported by University of California (UC) office of the President Grant LFR-18-548554. All opinions and statements reported here represent those of the authors.

REFERENCES

[1] C. F. A. A. Technology, "Connected and Automated Vehicles," 2017, accessed 2017 from http://autocaat.org/Technologies/Automated_and_Connected_Vehicles.

[2] O. of the Assistant Secretary for Research and T. I. J. Program, "Connected Vehicles," https://www.its.dot.gov/research_areas/connected_vehicle.htm.

[3] U. D. of Transportation, "The Connected Vehicle Reference Implementation Architecture," accessed March 2019 from https://local.iteris.com/cvria/html/applications/applications.html.

[4] "CV Pilot Deployment Program," accessed from https://www.its.dot.gov/pilots/cv_pilot_apps.htm.

[5] "USDOT: Security Credential Management System (SCMS)," accessed from https://www.its.dot.gov/factsheets/pdf/CV_SCMS.pdf.

[6] J. Harding, G. Powell, R. Yoon, J. Fikentscher, C. Doyle, D. Sade, M. Lukuc, J. Simons, J. Wang, *et al.*, "Vehicle-to-vehicle communications: readiness of V2V technology for application." United States. National Highway Traffic Safety Administration, Tech. Rep., 2014.

[7] F. Wang, Y. Xu, H. Zhang, Y. Zhang, and L. Zhu, "2flip: A two-factor lightweight privacy-preserving authentication scheme for vanet," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 2, pp. 896–911, 2016.

[8] A. Abdo, G. Wu, and N. Abu-Ghazaleh, "Secure ramp merging using blockchain," in *2021 IEEE Intelligent Vehicles Symposium (IV)*, 2021, pp. 401–408.

[9] N. Bißmeyer, C. Stresing, and K. M. Bayarou, "Intrusion detection in vanets through verification of vehicle movement data," in *2010 IEEE Vehicular Networking Conference*, 2010, pp. 166–173.

[10] Y. Fan, Y. Ye, and L. Chen, "Malicious sequential pattern mining for automatic malware detection," *Expert Systems with Applications*, vol. 52, pp. 16 – 25, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S095741741600004X

[11] "SAVIOR: Securing autonomous vehicles with robust physical invariants," in *29th USENIX Security Symposium (USENIX Security 20)*. Boston, MA: USENIX Association, Aug. 2020. [Online]. Available: https://www.usenix.org/conference/usenixsecurity20/presentation/quinonez

[12] J. B. Kenney, "Dedicated short-range communications (dsrc) standards in the united states," *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1162–1182, 2011.

[13] S. Sen and L. Head, "Controlled optimization of phases at an intersection," *Transportation Science*, vol. 31, pp. 5–17, 02 1997.

[14] Y. Feng, L. Head, S. Khoshmagham, and M. Zamanipour, "A real-time adaptive signal control in a connected vehicle environment," *Transportation Research Part C: Emerging Technologies*, vol. 55, 01 2015.

[15] Z. Wang, K. Han, B. Kim, G. Wu, and M. Barth, "Lookup table-based consensus algorithm for real-time longitudinal motion control of connected and automated vehicles," 07 2019.

[16] T. Stanger and L. Re, "A model predictive cooperative adaptive cruise control approach," 06 2013, pp. 1374–1379.

[17] J. Ploeg, B. T. M. Scheepers, E. van Nunen, N. van de Wouw, and H. Nijmeijer, "Design and experimental evaluation of cooperative adaptive cruise control," in *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2011.

[18] V. Milanés, S. E. Shladover, J. Spring, C. Nowakowski, H. Kawazoe, and M. Nakamura, "Cooperative adaptive cruise control in real traffic situations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 1, pp. 296–305, 2014.

[19] M. Amoozadeh, H. Deng, C. Chuah, M. Zhang, and D. Ghosal, "Platoon management with cooperative adaptive cruise control enabled by vanet," *Vehicular communications*, vol. 2, no. 2, pp. 110–123, 2015.

[20] E. Mazor, A. Averbuch, Y. Bar-Shalom, and J. Dayan, "Interacting multiple model methods in target tracking: a survey," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 34, no. 1, pp. 103–123, 1998.

[21] M. A. Skoglund, G. Hendeby, and D. Axehill, "Extended kalman filter modifications based on an optimization view point," in *2015 18th International Conference on Information Fusion (Fusion)*, 2015, pp. 1856–1861.

[22] C. Murguia and J. Ruths, "Cusum and chi-squared attack detection of compromised sensors," in *2016 IEEE Conference on Control Applications (CCA)*, 2016, pp. 474–480.

[23] T. T. Nguyen and V. J. Reddi, "Deep reinforcement learning for cyber security," *CoRR*, vol. abs/1906.05799, 2019. [Online]. Available: http://arxiv.org/abs/1906.05799

[24] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018.

[25] C. Shyalika, "A beginners guide to q-learning," *towardsdatascience*.

[26] M. M. Minderhoud and P. H. Bovy, "Extended time-to-collision measures for road traffic safety assessment," *Accident Analysis & Prevention*, vol. 33, no. 1, pp. 89–97, 2001.

[27] "VENTOS - VEhicular NeTwork Open Simulator," accessed from http://maniam.github.io/VENTOS/.

[28] C. Sommer, R. German, and F. Dressler, "Bidirectionally coupled network and road traffic simulation for improved ivc analysis," *IEEE Transactions on Mobile Computing*, vol. 10, no. 1, pp. 3–15, January 2011.

[29] "SUMO - Simulation of Urban Mobility," accessed from http://sumo.dlr.de/index.html.

[30] "OMNeT++," accessed from https://www.omnetpp.org/.

[31] R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein, "The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2118–2125.

[32] A. Abdo, S. M. B. Malek, Z. Qian, Q. Zhu, M. Barth, and N. Abu-Ghazaleh, "Application level attacks on connected vehicle protocols," in *22nd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2019)*. USENIX Association.

[33] S. Boddupalli, A. S. Rao, and S. Ray, "Resilient cooperative adaptive cruise control for autonomous vehicles using machine learning," *CoRR*, vol. abs/2103.10533, 2021. [Online]. Available: https://arxiv.org/abs/2103.10533