Threat Detection for Collaborative Adaptive Cruise Control in Connected Cars

Matthew Jagielski
Northeastern University

Nicholas Jones
Northeastern University

Chung-Wei Lin
Toyota InfoTechnology Center

Cristina Nita-Rotaru
Northeastern University

Shinichi Shiraishi
Toyota InfoTechnology Center

ABSTRACT

We study collaborative adaptive cruise control as a representative application for safety services provided by autonomous cars. We provide a detailed analysis of attacks that can be conducted by a motivated attacker targeting the collaborative adaptive cruise control algorithm, by influencing the acceleration reported by another car, or the local LIDAR and RADAR sensors. The attacks have a strong impact on passenger comfort, efficiency, and safety, with two of such attacks being able to cause crashes. We also present two detection methods rooted in physical-based constraints and machine learning algorithms. We show the effectiveness of these solutions through simulations and discuss their limitations.

ACM Reference Format:
Matthew Jagielski, Nicholas Jones, Chung-Wei Lin, Cristina Nita-Rotaru, and Shinichi Shiraishi. 2018. Threat Detection for Collaborative Adaptive Cruise Control in Connected Cars. In *WiSec '18: Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks, June 18–20, 2018, Stockholm, Sweden.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3212480.3212492

1 INTRODUCTION

Rapid developments in the last years have made autonomous and connected cars a reality. Such cars are equipped with sensors that allow them to send and receive signals, sense the physical environment around them, and interact with other vehicles or entities, including remote cloud services.

One of the major questions that must be answered for full adoption of the connected car paradigm is "How will connected cars and applications for connected cars be protected against cyber-attacks?" This is a difficult task given that connected cars are complex entities consisting of numerous hardware and software components having several unprotected or vulnerable access points such as on-board diagnostics (OBD) and multiple types of communication, with different levels of security.

Previous work has focused on in-car vulnerabilities, exploiting the lack of secure communication between sensors and their managing electronic control units (ECUs), or the lack of authentication for ECUs and secure communication on the controller area network (CAN). Some solutions have been proposed to individual attacks. For example, several works proposed authentication and integrity solutions for CAN communication [31] or for car-to-car communication [30]. Other works have considered intrusion detection systems (IDS) for CAN communication based on fingerprinting ECUs [9], or secure access to the CAN via a network controller [27]. Multiple corporations and industry consortiums have proposed in-car architectures that take security into consideration at design time. Examples include: AUTOSAR [25] and Evita [23].

One particularly important class of applications for connected cars are those providing increased safety, such as traffic and congestion control, collision avoidance, intersection management, and assisted-turn. Such applications rely on information received from sensors and other cars on the road to automatically make decisions. Several types of control algorithms have been proposed, based on information reported by preceding cars and local sensors or cameras. For example, in collaborative adaptive cruise control (CACC), cars are organized into platoons and each car adjusts its acceleration based on the acceleration of the preceding car, received via car-to-car communication, and information from its own RADAR and LIDAR sensors. Given the critical role of such applications, it is very important that they are resilient to attacks.

Previous work in CACC has mainly focused on attacks against different controllers [1, 10, 11, 13, 14, 28] by showing the impact of attacks against the network communication protocol on the efficiency of the platoon. Lying acceleration attacks have been identified in [1]. Attacks using malfunctioning RADAR and LIDAR were shown in [28], but with extreme values (i.e. $30m/s^2$ acceleration or $11m$ distance) which would be readily identified by simple defenses. Neither of [1] and [28] proposed defenses.

In this paper we study CACC and consider attacks caused by attackers who compromise a car and use it to attack another victim, or compromise a subset of the victim's car (RADAR or LIDAR sensors) and use it to control the acceleration of the victim. Unlike previous work we propose attacks that are harder to detect (maxima of $5m/s^2$ for acceleration or $3m$ for distance), and we also discuss defenses. Specifically:

• We identify four attacks ACL, VEL, POS, and VEL-POS that can be conducted by an attacker that has compromised either the acceleration dissemination subsystem, the RADAR or the LIDAR sensors. These attacks impact the safety, passenger comfort, and efficiency of CACC. We show the impact of these attacks through simulations: the ACL attack has the strongest impact on passenger comfort, the VEL attack is the most effective against efficiency, and the POS and VEL-POS attacks both result in a crash.

WiSec '18, June 18–20, 2018, Stockholm, Sweden

ACM ISBN 978-1-4503-5731-9/18/06…$15.00
https://doi.org/10.1145/3212480.3212492

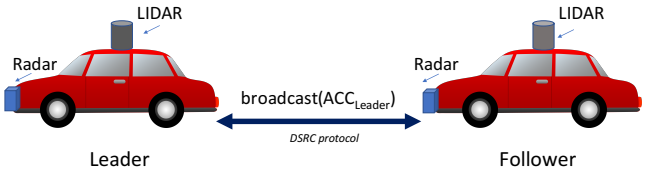


Figure 1: CACC overview.

• We propose two solutions: PHY – rooted in cyber-physical properties – enforces constraints derived from kinematics equations and HMM – drawn from anomaly detection – uses hidden Markov models. We demonstrate the defense mechanisms through simulations. Our results show that while simple and fast, PHY can not detect all considered attacks, including attacks causing crashes, while HMM can detect all attacks we considered.

Concurrent work [22] considers position lying attacks, in a model where the adaptive cruise control algorithm uses messages from more than one car, proposing defenses that leverage voting. In our model, the only information available to the adaptive cruise control algorithm is acceleration from the preceding car, and RADAR and LIDAR information from own sensors.

2 CACC

Collaborative Adaptive Cruise Control (CACC) extends traditional Adaptive Cruise Control (ACC) by involving the preceding car in the acceleration computation. Different methods have been proposed for providing this application, using different algorithms, different sensors, and different sensor placements.

We consider the CACC algorithm described in [2] and showed in Figure 1, which is used for platoon management of cars. Each car is equipped with dedicated short-range communications (DSRC) communication [16], which allows it to receive acceleration information from the preceding car, and with its own RADAR and LIDAR sensors, that allow it to measure the velocity and position of the preceding car. The acceleration of the preceding car is sent using the DSRC protocol and its basic safety messages (BSMs). (DSRC provides authentication and integrity of the communication and its availability in cars is increasing [15, 29].)

Each vehicle in the platoon tries to maintain a safe space-gap with its preceding vehicle. Safe space-gap, denoted by g_{safe}, is determined by speed and maximum deceleration ability of the vehicle (v and D^{max}) and those of the preceding vehicle (v_p and ${D_p}^{max}$) , and is given by

$$g_{safe} = v * 0.1 + \frac{v^2}{2D^{max}} - \frac{{v_p}^2}{2{D_p}^{max}} + 1.0$$

where 1.0 is the minimum space-gap. As soon as the instantaneous space-gap $g \leq g_{safe}$, the vehicle switches to collision avoidance mode and uses maximum deceleration D^{max} to avoid collision. Acceleration is computed as:

$$a_{t+1} = K_a a_t + K_v(v_p - v) + K_g(g - G_{min} - vT_g)$$

where a_{t+1} is next timestep's acceleration, a_t is current acceleration, K_a is an acceleration constant, K_v is a velocity constant, K_g is a position constant, v_p is preceding car's velocity, v is car's velocity, g is current gap, G_{min} is minimum gap, and T_g is safe gap. ($K_a = 0.66, K_v = 0.99s^{-1}, K_g = 4.08s^{-2}, G_{min} = 2m, T_g = 0.55s$ as specified in [2].)

The CACC algorithm has three main goals: safety, efficiency, and passenger comfort captured by metrics described below.

Safety: Given a minimum safe *time*-gap g^t_{safe} (computed as the safe space gap divided by velocity), we will call a given timestep of a run of CACC *safe* if the time-gap g^t_i between each pair of consecutive cars $i, i+1$ is at least g^t_{safe}. The whole run will be safe if it is safe at every time-step, T_j. The severity of a safety violation is captured by the proportion of g^t_{safe} by which the gap is violated:

$$crash = \max_{T_j} \left\{ 0, \max_i \frac{g^t_{safe} - g^t_i}{g^t_{safe}} \right\}.$$

The maximum value this can take is 1, at which point two cars have crashed. The minimum, when all gaps are safe, is 0.

Efficiency: The goal of CACC is to build a platoon of cars which are traveling with as little distance as possible between them. In order to be efficient a run needs to minimize the value for car i of

$$waste_i = \int_{t=0}^{t_{end}} \left(g^t_i - g^t_{safe} \right) dt.$$

This integral is computed until the end of the run t_{end}, and total waste is the maximum waste on a car.

Passenger comfort. When safety and correctness are accounted for, the algorithm also needs to provide for passengers' comfort. A common way to quantify passenger comfort is using the magnitude of *jerk*, the third derivative of position:

$$jerk = \frac{da}{dt}.$$

We will consider the maximum jerk over all timesteps in order to measure passenger comfort over a run of CACC.

3 ATTACKS AGAINST CACC

The considered CACC algorithm running on a car relies on acceleration from other cars and its own RADAR and LIDAR sensors in order to compute the new acceleration. Based on the information they can manipulate we consider two types of attackers.

Influence acceleration of other than the victim's car. Information between cars is sent over DSRC, which provides authentication and integrity, thus outside adversaries cannot compromise this information unless they break the cryptographic assumptions of the protocol. Outside adversaries can also disturb the communication at the physical layer [4]. We do not consider such attacks. Instead, we consider an attacker who has compromised a car other than the victim's car and sends incorrect acceleration packets to influence the computation (by the CACC subsystem) of the victim's acceleration. This is a weaker threat model since the attacker may compromise any car and then just place it near the victim. This attack will propagate through the platoon of cars allowing the attacker to be not only immediately preceding the victim but also many cars forward. We consider the impact of only one attacker.

Influence RADAR and/or LIDAR sensors of the victim. Information from RADAR and LIDAR sensors are obtained from local sensors and they provide velocity and position information about the preceding car. Attackers can manipulate data from the victim car's

sensors, either directly, by compromising a subset of the victim car, or indirectly, by remotely manipulating the sensor's physical layer signals [7, 21]. We assume that the attacker did not totally compromise the victim's car, but has control over just the LIDAR, just the RADAR, and over both LIDAR and RADAR. We distinguish RADAR-only attacks and LIDAR-only attacks since these require different capabilities, especially to conduct remotely [7, 21]. This is a stronger attack as it requires direct control over sensors in the victim car or the ability to manipulate physical layer sensor signals.

We identify four specific attacks defined by the way we model lying about information and the system goals impacted the most.

ACL attack. This attack occurs when the attacker can modify acceleration packets. We consider attacks that are not easily detectable, i.e. using values with acceptable ranges for the CACC subsystem, for the subsystem we consider these values lie within $-5m/s^2$ to $5m/s^2$ [2]. While all kind of lying patterns are possible, we focus on patterns that can generate oscillations because due to the small range of changes, we do not expect the impact on safety to be significant. We consider attacks modeled by a sinusoid function where the lying acceleration a_{fake} is defined by the following:

$$a_{fake} = a_{true} + c_a \sin(ft)$$

where f is the frequency of lying, t is time, and c_a is the magnitude. The system goal impacted is passenger comfort.

VEL attack. This attack takes place when the attacker can modify RADAR (velocity) sensor values. Velocity manipulation has bigger impact than ACL, as the CACC algorithm puts more weight on velocity values and velocity is more difficult to bound. We consider attacks not easily detectable that cause inefficiency by slowly decreasing the velocity measurement of the preceding car causing the victim to slow down to match. While many functions are possible, we consider attacks modeled as a linear function represented by the following equation:

$$v_{fake} = v_{true} - c_v\, t$$

where c_v is the magnitude of the lying, t is time, and v_{fake} and v_{true} are the induced speed and the true speed measured by the victim. There is a tradeoff in choosing c_v – a large value will create a powerful attack, but make the attack easy to detect. A small value will be harder to detect, but have less impact. The system goal impacted is efficiency.

POS attack. This attack occurs when the attacker has the ability to modify LIDAR (position) sensor values. This attacker is able to decrease the safety of the algorithm, increasing the likelihood of a crash. It operates by slowly increasing the distance measured to the direct leader so that the follower will overestimate the gap and follow too closely. While many functions are possible, we model the attack as a linear function given by the following equation:

$$d_{fake} = d_{true} + c_d\, t$$

where c_d is the magnitude of the lying, set by the attacker to balance probability of detection with the power of the attack, t is time, and d_{fake} and d_{true} are the distance from origin as induced by the attack and in truth, respectively, as measured by the victim. System goal impacted is safety.

VEL-POS attack. The fourth and final attack we consider is when the attacker can modify both LIDAR (position) and RADAR (velocity) sensor values. This adversary is strictly more powerful than the POS adversary, so it can also decrease safety. While other functions are possible, we consider functions where the attacker slowly increasing both sensor values, as represented by the following equations:

$$v_{fake} = v_{true} + c_v\, t,\ d_{fake} = d_{true} + c_d\, t$$

where the c_v and c_d are the magnitude of lying for speed and distance, t is time, and d_{fake} and d_{true} are the distance from origin as induced by the attack and in truth, while v_{fake} and v_{true} are the induced speed and the true speed, all measured by the victim. System goal impacted is safety.

4 ATTACKS MITIGATION

The essence of the attacks we consider is the ability of an attacker to lie about information used to compute the victim's acceleration. While detecting such attacks is a very difficult problem, there are unique properties of CACC that make the problem more tractable. First, the information reported by attackers in CACC captures cyber-physical properties and must obey the kinematic equations. Second, the information is reported periodically, thus one can build models and observe their behavior over time. Below we describe two mitigation techniques that leverage these properties.

Physics-Based (PHY). Each car uses sensors to measure the position and velocity of preceding car, and receives packets containing the acceleration value of preceding car. Since cars are physical objects, their behavior in terms of position, velocity, and acceleration must follow certain well defined laws of kinematics. By using these laws, we can detect inconsistencies between these values as a result of an attack. Let t_d be the length of time between measurements, p_{old}, p_{new} be the old and new position values of the direct leader, v_{old}, v_{new} be the old and new velocities, and a_{old}, a_{new} be the accelerations received via DSRC from the direct leader. Write $v_{min} = \min(v_{old}, v_{new}), v_{max} = \max(v_{old}, v_{new}), a_{max} = \max(a_{new}, a_{old}), a_{min} = \min(a_{new}, a_{old})$. We introduce parameters, ϵ_p and ϵ_v, to account for measurement noise and any time difference between the. The values of ϵ_p and ϵ_v can be tuned before deploying the defense. On a small enough time step, the following equations *must* be true, at the car performing the measurements:

$$v_{min}\, t_d + 0.5\, a_{min}\, t_d^2 - \epsilon_p \le p_{new} - p_{old}$$

$$p_{new} - p_{old} \le v_{max}\, t_d + 0.5\, a_{max}\, t_d^2 + \epsilon_p$$

$$a_{min}\, t_d - \epsilon_v \le v_{new} - v_{old} \le a_{max}\, t_d + \epsilon_v$$

A violation of any of these inequalities is flagged as an attack.

This method has the advantage that is has very small computational overhead and requires very little storage – only the old and new values of the position, velocity, and acceleration. It can easily be deployed on a car.

Threats to validity. While PHY is suited for scenarios where the attacker controls only a single metric (i.e. position only), it does not work well for sophisticated attackers that can control two or more metrics (i.e. position and velocity). Such an attacker may be able to change the metrics in a consistent manner to mimic cyber-physical properties, matching thus the detection equations and avoiding detection. As any threshold-based method, PHY is also vulnerable to attacks just below the detection threshold.

Hidden Markov Model (HMM). For scenarios where an attacker controls more than one metric, we leverage the strong temporal component of CACC – position, velocity, and acceleration of a

car and its leader all influence these values later in the protocol. We use a Hidden Markov Model, an anomaly detection mechanism, to fit the time series data of CACC and learn temporal dependencies.

An HMM can be described as a set of n states Q, an $n \times n$ transition probability matrix A, emission probabilities θ. We use two states, as we observe two types of behavior in the CACC algorithm – a synchronization phase where cars create the safe gaps, and a stable phase, where cars stay at a roughly fixed velocity. The i, jth entry of A represents the probability that, given that state i is observed at time $t-1$, the state j will be observed at t. The emission probabilities θ represent the probability distributions of values taken at each state. In our case, the emission probabilities represent the distribution of position, velocity, and acceleration values for each state. We can use a dynamic programming solution called the forward algorithm [24] to compute a log-likelihood that a time series was generated by a given HMM and detect anomalies by using a threshold ρ_h.

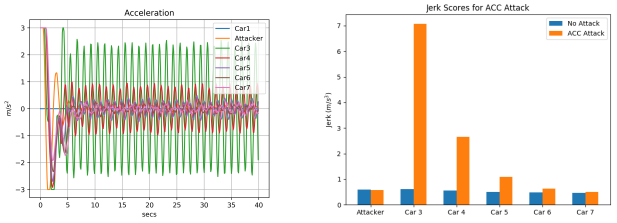
While HMM can address attackers that control multiple metrics, it is more complex than PHY. The dynamic programming solution for probability computation, the forward algorithm, runs in time $O(n^2 t)$, where t is the number of time steps in the time series considered, and n is the number of states. It also requires $O(n + t)$ space, more than the constant amount required in PHY.

Threats to validity. Given knowledge of the parameters of an HMM, it is possible to construct attacks on the model, either attempting to bypass the detection (evasion), or attempting to influence the training to make the model less reliable (poisoning). For example, such a misclassification for speech (a different setting than our application) given access to the HMM parameters, was shown in [6]. While not trivial, a similar attack could be performed for our application by constructing an optimization problem to maximize impact under the constraint that the strategy is not flagged as an anomaly. This is a well known weakness of machine learning models in general and on-going work in adversarial machine learning is trying to address it. One possible solution is to use ensembling of models to provide some resilience to both evasion and poisoning attacks [3, 5]. In our case, an ensemble of PHY and HMM, or an ensemble of models constructed from disparate vantage points (where they would be difficult to manipulate simultaneously), would improve the robustness of our proposed defense.

5 EXPERIMENTAL RESULTS

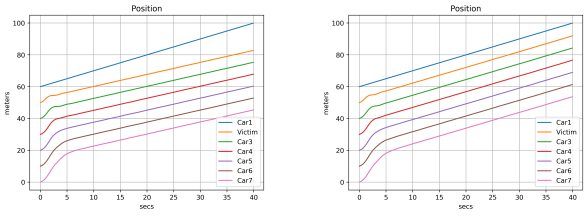
Methodology. We implemented our own simulator in Python, simulation is discrete, a run is 400 steps, where each step is 0.1s and was chosen to match the frequency of acceleration messages. We simulate a platoon of 7 cars, car length is $5m$. The cars start at $1m/s$ with a distance between cars of $10m$. We model sensor measurement error with Gaussian noise, with standard deviation of $3cm$ [19] for LIDAR and $0.1m/s$ [12] for RADAR. We use the CACC algorithm from Section 2. The minimum safe-gap is $0.55s$ [2], with a 2m leeway, resulting in a 2.55m gap (or 7.55m from front to front including car length). The maximum deceleration is $5m/s^2$.

The attacker implements the attacks described in Section 3. We use only one attacker in the platoon, For the ACL attack, the victim is Car 3 and the attacker is Car 2. For the other attacks, the compromised car and victim are the same – Car 2. A distance of $5m$ between two adjacent cars indicates a crash has occurred.



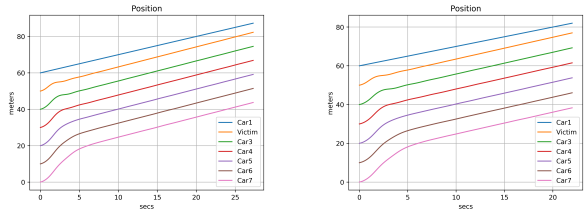
(a) Acceleration (b) Jerk

Figure 2: Impact of ACL attack ($c_a = 5$ and $f = 5$).



(a) $c_v = 1$ (b) $c_v = 0.1$

Figure 3: Impact of VEL attack on positions within the platoon. Note the increasing gap between Car 1 and the victim.



(a) POS attack ($c_d = 0.1$) (b) VEL-POS attack ($c_v = 0.2$ and $c_d = 0.1$)

Figure 4: Impact of POS and VEL-POS attacks. In both cases the attack causes a crash (indicated by distance of $5m$) at 27.6 seconds (left) and 21.2 seconds (right) into the run.

5.1 Attacks

We show several instances of our attacks and a summary in Table 1). We report averages over 10 runs.

ACL attack. This adversary, with the goal of introducing instability, adds a sinusoidal value ($c_a = 5m/s^2$ and $f = 5Hz$) to its accelerations as shown in Figure 2(a). The corresponding jerk is shown in Figure 2(b). The average magnitude of the jerk experienced by Car 3 is 7.07 m/s^3, compared to an average jerk of magnitude 0.56 m/s^3 when there is no attack. Even Car 6 is affected – its average magnitude of jerk when under attack is 0.61 m/s^3 while its jerk has an average magnitude of 0.43 m/s^3 in an benign environment. We find that the attack also impacts Car 3's efficiency, the average time gap is 6.22 seconds for Car 3 when under attack, but is only 2.10 seconds when not under attack.

VEL attack. The VEL adversary slowly decreases the RADAR velocity measurements of the car in front of it, leading the victim to believe its direct leader is going slower than it really is. As a result, the victim slows down to compensate, increasing the gap.

Attack	Jerk (m/s^3)	Waste (s)	Crash
No attack	0.56	2.10	0
ACL ($c_a = 5$, $f = 5$)	7.07	3.14	0
VEL ($c_v = 1$)	0.59	9.32	0
POS ($c_d = 0.1$)	0.73	0.69	crash
VEL-POS ($c_v = 0.2$, $c_d = 0.1$)	0.86	0.60	crash

Table 1: Comfort, efficiency, and safety metrics for victim.

We experiment with two different values of c_v, 0.1 and 1. When $c_v = 1$, as seen in Figure 3(a), this attack significantly impacts the efficiency at Car 2. The mean gap between the victim and Car 1 is 9.31 seconds, compared to 2.10 seconds with no attack. This is an increase of 343%. The jerk in this case is still 0.60 m/s^3, not significantly different from 0.56 m/s^3 without any attack. With $c_v = 0.1$, as shown in Figure 3(b), there is still some efficiency impact. The mean gap in this case is 2.37 seconds, a 13% increase over no attack. There is still little stability change – the maximum jerk is only 0.61 m/s^3.

POS attack. This adversary manipulates the victim's LIDAR reading to slowly increase the distance measured to the direct leader. This will cause the victim to decrease the gap. We set the parameter for the attack to match the magnitude in error for LIDAR sensors, we set $c_d = 0.1$. The key impact of the POS attack is that it is able to cause the victim, Car 2, to crash. This is illustrated in Figure 4(a). After an average of 27.6 seconds, the magnitude of the error is large enough to cause a crash (distance of $5m$ vs. $7.55m$ in stable state). The crash is induced with a fairly small impact in performance and stability, the victim 's average jerk magnitude is only 0.70 m/s^3, compared to 0.56 m/s^3 without the attack. This may be an indication that the attack will be difficult to detect.

VEL-POS attack. This adversary combines the power of the VEL and POS adversaries, we select comparable values for the parameters for VEL and POS, we use $c_v = 0.2$ and $c_d = 0.1$.

The VEL-POS attack is a strict improvement over POS, so we expect it to cause crashes – we see this happening in Figure 4(b). While POS caused crashes after an average of 27.6 seconds, VEL-POS is able to cause crashes after an average of 21.2 seconds. Similar to POS, this is done with limited impact on performance and stability.

5.2 Defenses

Summary of detection rate is shown in Table 2 over 10 runs, of 400 steps each. The detection rate is defined as number of successful detections divided by the number of times detection algorithm is invoked. PHY is invoked 400 times (at each step, or every 0.1s), for HMM this number is 20 (every 50 steps, or 5 seconds).

PHY defense. We configure PHY with tolerance parameters $\epsilon_v = 0.4m/s$ and $\epsilon_p = 0.15m$. Figure 5(a) shows the ACL attack ($c_a = 5$, $f = 5$), with the PHY defense in place. The vertical bar indicates that an attack was detected and as seen from the graphs this attack is detected at several timesteps. We next applied PHY to three instances of VEL attack ($c_v = 0.05$, $c_v = 0.1$, and $c_v = 1$) and to the POS and VEL-POS attacks from Figure 4. We show only VEL attack with $c_v = 1$ and $c_v = 0.05$ in Figures and 5(b) and 5(c). While PHY detects the VEL attack ($c_v = 1$), it can not detect any of the other attacks including the ones causing crashes. Table 2 shows that the only attack for which PHY has consistently good performance is the VEL attack ($c_v = 1$).

Attack	PHY	HMM
No attack (false positives)	0.35%	1.5%
ACL ($c_a = 5$, $f = 5$)	25.75%	77.5%
VEL ($c_v = 1$)	95.13%	83.5%
VEL ($c_v = 0.1$)	0.58%	79.5%
VEL ($c_v = 0.05$)	0.45%	79.5%
POS ($c_d = 0.1$)	0.25%	74%
VEL-POS ($c_v = 0.2$, $c_d = 0.1$)	0.13%	90%

Table 2: Detection rate for all attacks (over 10 runs).

HMM defense. We set the threshold for anomaly detection to minimize false positives, $\rho_h = -25$. We test HMM against all attack instances shown in Figures 2,3,4 and 5(c). HMM is successful at detecting all of them. Notably, the POS attack is successfully identified by HMM well before it causes a crash (see Figure 6(a)). We are also able to identify the VEL-POS attack before it causes a crash (see Figure 6(b)). We also show in Figure 6(c), that HMM correctly identifies that there is no attack in the benign scenario. Table 2 shows HMM has good detection accuracy for all considered attacks, being the most effective against VEL-POS.

6 RELATED WORK

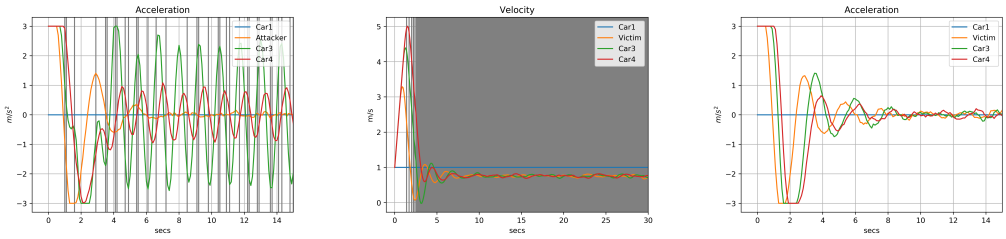
In-car. Previous work studied in-car vulnerabilities and defenses. Attacks examples include taking control of the brakes and steering of a vehicle [8, 17, 20], influencing the tire-pressure monitoring system (TPMS) [26] and proposed defenses include authentication and integrity for CAN [31] or for car-to-car communication [30], intrusion detection system [9] or secure access for CAN [27].

Connected cars. Most of the work for connected cars to date has focused on attacks. In particular, attacks have been shown to be effective against DSRC, the MAC protocol for car-to-car communications [18], and demonstration of physical against sensors like RADAR [7] and LIDAR [21].

CACC. Attacks against CACC controllers have also been done in recent works [1, 10, 11, 13, 14, 28] exploiting the network communication protocol, jamming the physical layer, or using limitations of the public key infrastructure for DSRC. The difference in the algorithms for CACC lies in the sensors assumed by the algorithm, their positioning on the car, and the control algorithms themselves. Most of the work has focused on attacks manifested through acceleration or deceleration of other cars. Our work is closest to the work in [1] which considers the same car sensors and algorithm we use and the work in [28] which considers sensor malfunctioning.

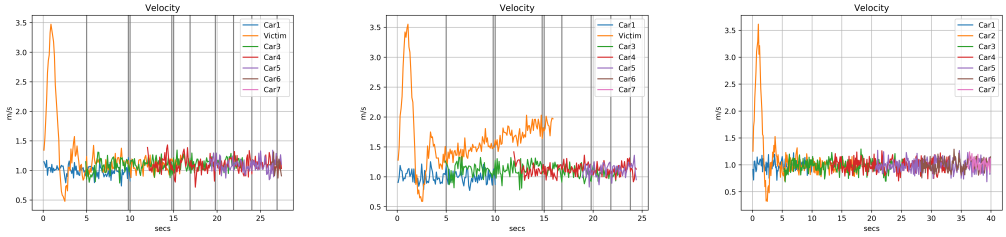
7 CONCLUSION

We demonstrate attacks against cooperative adaptive cruise control applications that use acceleration reported via car-to-car communication, and distance and velocity measured by LIDAR and RADAR sensors, respectively. We show that the attacks have an impact on safety, efficiency, and passenger comfort, with two attacks causing crashes. We also propose an HMM-based mitigation technique that can successfully detect the specific attacks we considered.



(a) ACL attack ($c_a = 5, f = 5$) (b) VEL attack ($c_v = 1$) (c) VEL attack ($c_v = 0.05$)

Figure 5: PHY defense ($\epsilon_v = 0.4m/s$ and $\epsilon_p = 0.15m$). Detected attacks are shown with a gray line. PHY detects ACL and VEL ($c_v = 1$) attacks. PHY does not detect the VEL attack ($c_v = 0.05$).



(a) POS attack (b) VEL-POS attack (c) No attack

Figure 6: HMM defense ($\rho_h = -25$). Detected attacks are shown with a gray line. HMM detects the POS and VEL-POS attacks from Figure 4 before the crash and correctly indicates no attack in benign case.

REFERENCES

[1] Mani Amoozadeh, Arun Raghuramu, Chen-Nee Chuah, Dipak Ghosal, H. Michael Zhang, Jeff Rowe, and Karl Levitt. 2015. Security Vulnerabilities of Connected Vehicle Streams and Their Impact on Cooperative Driving. *IEEE Communications Magazine* 53 (June 2015). Issue 6.

[2] Mani Amoozadeha, Hui Dengb, and H. Michael Zhangb Chen-Nee Chuaha, and Dipak Ghosalc. 2015. Platoon Management with Cooperative Adaptive Cruise Control Enabled by VANET. *Veh. Commun.* 2, 2 (April 2015), 110–123.

[3] Davide Ariu, Roberto Tronci, and Giorgio Giacinto. 2011. HMMPayl: An intrusion detection system based on Hidden Markov Models. *Computers & Security* 30, 4 (2011), 221–241.

[4] Yuksel Ozan Basciftci, Fangzhou Chen, Joshua Weston, Ron Burton, and Can Emre Koksal. 2015. How Vulnerable Is Vehicular Communication to Physical Layer Jamming Attacks?. In *VTC Fall*. IEEE, 1–5.

[5] Battista Biggio, Igino Corona, Giorgio Fumera, Giorgio Giacinto, and Fabio Roli. 2011. Bagging classifiers for fighting poisoning attacks in adversarial classification tasks. In *International workshop on multiple classifier systems*.

[6] Nicholas Carlini, Pratyush Mishra, Tavish Vaidya, Yuankai Zhang, Micah Sherr, Clay Shields, and David Wagner. 2016. Hidden Voice Command.. In *USENIX Security*.

[7] R. Chauhan, R. M. Gerdes, and K. Heaslip. 2014. Demonstration of a False-data Injection Attack Against an FMCW Radar. In *ESCAR*.

[8] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, and Tadayoshi Kohno. 2011. Comprehensive Experimental Analyses of Automotive Attack Surfaces. In *Proceedings of the 20th USENIX Conference on Security*.

[9] Kyong-Tak Cho and Kang G. Shin. 2016. Fingerprinting Electronic Control Units for Vehicle Intrusion Detection. In *USENIX Security*. 911–927.

[10] Soodeh Dadras, Ryan M. Gerdes, and Rajnikant Sharma. 2015. Vehicular Platooning in an Adversarial Environment. In *ASIA CCS*. 167–178.

[11] Bruce DeBruhl, Sean Weerakkody, Bruno Sinopoli, and Patrick Tague. 2015. Is Your Commute Driving You Crazy?: A Study of Misbehavior in Vehicular Platoons. In *ACM WiSec*.

[12] DICKEY-john. 2018. DICKEY-john. http://www.dickey-john.com/product/radar-iii/

[13] R. M. Gerdes, C. Winstead, and K. Heaslip. 2013. Cps: an efficiency-motivated attack against autonomous vehicular transportation. In *ACSAC*.

[14] J. J. Haas. 2009. The effects of wireless jamming on vehicle platooning.

[15] Andrew J. Hawkins. 2017. Cadillac's [CTS] sedans can now "talk" to each other, which may make driving way less deadly. https://www.theverge.com/2017/3/9/14869110/cadillac-cts-sedan-v2v-communication-dsrc-gm

[16] John B. Kenney. 2011. Dedicated Short-Range Communications (DSRC) Standards in the United States. *IEEE* 99 (2011). Issue 7.

[17] Karl Koscher, Alexei Czeskis, Franziska Roesner, Shwetak Patel, Tadayoshi Kohno, Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, and Stefan Savage. 2010. Experimental Security Analysis of a Modern Automobile. In *IEEE Symposium on Security and Privacy*. 447–462.

[18] Christine Laurendeau and Michel Barbeau. 2006. *Threats to Security in DSRC/WAVE*. 266–279. https://doi.org/10.1007/11814764_22

[19] Velodyne Lidar. [n. d.]. Velodyne Lidar. http://www.velodynelidar.com/downloads.html

[20] Charlie Miller and Chris Valasek. 2014. A Survey of Remote Automotive Attack Surfaces. In *Blackhat 2014*.

[21] J. Petit, B. Stottelaar, M. Feiri, and F. Kargl. 2015. Remote Attacks on Automated Vehicles Sensors: Experiments on Camera and LiDAR. In *Black-hat Europe*.

[22] Alberto Petrillo, Antonio Pescapé, and Stefania Santini. 2018. A collaborative approach for improving the security of vehicular scenarios: The case of platooning. *Computer Communications* 122 (2018), 59 – 75.

[23] EVITA Project. 2008. The EVITA Project: E-safety vehicle intrusion protected applications. https://www.evita-project.org/

[24] Lawrence R Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* 77, 2 (1989), 257–286.

[25] RENESAS. 2010. AUTOSAR. https://www.renesas.com/en-us/solutions/automotive/technology/autosar.html

[26] Ishtiaq Rouf, Rob Miller, Hossen Mustafa, Travis Taylor, Sangho Oh, Wenyuan Xu, Marco Gruteser, Wade Trappe, and Ivan Seskar. 2010. Security and Privacy Vulnerabilities of In-car Wireless Networks: A Tire Pressure Monitoring System Case Study. In *USENIX Security*. 1.

[27] Shanker Shreejith and Suhaib A. Fahmy. 2015. Security Aware Network Controllers for Next Generation Automotive Embedded Systems. In *Proceedings of the 52Nd Annual Design Automation Conference (DAC '15)*. Article 39, 6 pages.

[28] Rens van der Heijden, Thomas Lukaseder, and Frank Kargl. 2017. Analyzing Attacks on Cooperative Adaptive Cruise Control (CACC). In *IEEE VNC*.

[29] Eric Waltz. 2018. Toyota and Lexus to Launch Tech to Connect Vehicles and Infrastructure in the U.S. in 2021. http://www.futurecar.com/article-2156-1.html

[30] William Whyte, André Weimerskirch, Virendra Kumar, and Thorsten Hehn. 2013. A security credential management system for V2V communications. In *2013 IEEE Vehicular Networking Conference, Boston, MA, USA, December 16-18, 2013*. 1–8.

[31] Miao Xu, Wenyuan Xu, Jesse Walker, and Benjamin Moore. 2013. Lightweight Secure Communication Protocols for In-vehicle Sensor Networks. In *CyCAR*. 12.