Blockchain-Integrated Multiagent Deep Reinforcement Learning for Securing Cooperative Adaptive Cruise Control

Gunasekaran Raja, *Senior Member, IEEE*, Kottilingam Kottursamy, Kapal Dev, *Senior Member, IEEE*, Renuka Narayanan, Ashmitha Raja, and K. Bhavani Venkata Karthik

***Abstract*— Connected and Autonomous Vehicles (CAVs) are an emerging solution to the issues of safe and sustainable transportation systems in the future. One major transport technology for CAVs is Cooperative Adaptive Cruise Control (CACC), for which unsignalized autonomous intersection crossing is a growing use case. CACC relies heavily on inter-vehicular communication and is thus vulnerable to message forgery and jamming attacks. Most solutions for CACC focus exclusively on enhancing efficiency or security but do not offer an integrated framework for achieving both on a large scale. In this paper, we propose a Blockchain-integrated Multi-Agent Deep Reinforcement Learning (Block-MADRL) architecture for enhancing the efficiency of CACC while cooperatively detecting attacks, reducing the fuel efficiency of identified attackers and securely notifying the overall network. Our approach uses multi-agent deep reinforcement learning to find fuel and throughput optimizing solutions for CACC and a cooperative verification mechanism based on Extended Isolation Forest (EIF) for attack detection. Attacker data is securely stored in a Road Side Unit (RSU) level blockchain, and we design a low-latency, high throughput consensus protocol for speedy and secure data dissemination. Simulation results indicate over 29.5% better lane throughput with our approach during acceleration forgery attack, up to 23% induced reduction in fuel efficiency of malicious vehicles, 17.6% higher blockchain throughput through our consensus protocol and over 8% improvement in attack detection rate compared to the state-of-the-art.**

***Index Terms*— Connected and autonomous vehicles, cooperative adaptive cruise control, Zyzzyva, blockchain, multi-agent deep reinforcement learning.**

I. Introduction

INTELLIGENT Transportation Systems (ITS) technologies that improve traffic flow stability, increase fuel efficiency and optimize safety and security have increased interest in recent years [1]. Of these technologies, Cooperative Adaptive Cruise Control (CACC) is a promising technique to improve safety and fuel efficiency by extending the prior Adaptive Cruise Control (ACC) technique with information exchange through Vehicle-to-Infrastructure (V2I) and Vehicle-to-Vehicle (V2V) communications.

In the years following the introduction of CACC as a concept, there were several practical and successful implementations held to prove its efficiency. Apart from the use of CACC in truck platooning, researchers have proposed its use in eco-driving on signalized corridors.

CACC has also been proposed for performing unsignalized and signalized intersection crossing [2]. A significant issue is that unsignalized intersections are more heavily reliant on inter-vehicle co-operation than signalized ones [3]. As a result, securing and enhancing the efficiency of CACC is of paramount importance in this scenario. One of the significant sources of security issues that CACC suffers from is its dependency on inter-vehicular communications [4]. The feasibility of CACC is directly dependent on both the wireless channel characteristics and the validity of the claimed accelerations reported by preceding vehicles [5]. This opens the mechanism to vulnerability during jamming and acceleration forgery attacks. CACC is also impacted by situations where the attacker has managed to damage the victim vehicle's lidar sensors, such that the velocity or position parameters are erratically perceived [31], [32]. Considering these concerns, it is necessary to develop an overall framework that not only optimizes the CACC goals of throughput, fuel and emission reduction but also provides integrated security against unpredictable acceleration forgery attacks and lidar attacks. The system must also be capable of fault tolerance and operation under real-time latency requirements, given the time-sensitive nature of cruise control.

The existing state-of-the-art solutions for CACC have started to focus on the benefits of Deep Reinforcement Learning – particularly the Deep Deterministic Policy Gradient (DDPG) algorithm [6] and multi-agent reinforcement learning approaches [7]. As of 2017, the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) algorithm [8] was introduced and made waves in the research community, with over 1025 citations to date. While this algorithm has been applied extensively in signalized traffic control, it is yet to be

Manuscript received 17 April 2021; revised 11 December 2021 and 13 March 2022; accepted 6 April 2022. Date of publication 29 April 2022; date of current version 8 July 2022. The work of Gunasekaran Raja, Renuka Narayanan, Ashmitha Raja, and K. Bhavani Venkata Karthik was supported by the NGNLab, Department of Computer Technology, Anna University, Chennai, India. The Associate Editor for this article was S. Wan. *(Corresponding author: Gunasekaran Raja.)*

Gunasekaran Raja, Renuka Narayanan, Ashmitha Raja, and K. Bhavani Venkata Karthik are with the NGNLab, Department of Computer Technology, Anna University, Chennai 600025, India (e-mail: dr.r.gunasekaran@ieee.org; sumikanth@gmail.com; ashmitharaja007@gmail.com; kvsaikarthik1906@gmail.com).

Kottilingam Kottursamy is with the School of Computing, SRM Institute of Science and Technology, Chennai 603203, India (e-mail: kottilik@srmist.edu.in).

Kapal Dev is with the Department of Institute of Intelligent Systems, University of Johannesburg, Johannesburg 2006, South Africa (e-mail: kapal.dev@ieee.org).

Digital Object Identifier 10.1109/TITS.2022.3168486

deployed for enhancing the efficiency of CACC at unsignalized intersections. We use MADDPG to learn a throughput and fuel optimizing bias to be added to the standard CACC desired acceleration.

We note that blockchain technology [9], [10] and decentralized Intrusion Detection Systems [25], [28] have been increasingly deployed in securing vehicular applications. In a recent development, the use of blockchain was also proposed for securing CACC against malicious messages, by recording the identities of untrusted vehicles on a Road Side Unit (RSU) level [12]. However, this RSU level blockchain was found to suffer from high latencies due to the Proof-of-Work consensus mechanism deployed. The authors also failed to specify mechanisms for handling fault tolerance. In our work, we propose a hybrid Zyzzyva-Joint-Proof-of-Work-Proof-of-Stake consensus mechanism for the RSU level blockchain to secure CACC. The proposed mechanism is designed to increase blockchain throughput compared to traditional longest fork consensus, and achieve fault tolerance at lower latencies using speculative methods.

We handle the detection of acceleration forgery and lidar sensor attacks via a neighbourhood based cooperative verification mechanism. Vehicles take turns to verify safety messages within their range by checking the difference between reported values and sensor-based estimates. The computed differences are stored as vectors and sent to the nearest RSU. This RSU then passes the difference vector through a low complexity Extended Isolation Forest (EIF). The EIF uses the vector to judge the likelihood of the vehicle performing a forgery or a lidar attack. This EIF approach is found to outperform existing state-of-the-art Hidden Markov Model (HMM) and kinematics based defence mechanisms on four stealthy attacks specified in [5].

In a nutshell, the key contributions of this work are:

1) A Blockchain-Integrated Multi-Agent Deep Reinforcement Learning (Block-MADRL) architecture for securing and enhancing the efficiency of large-scale unsignalized intersection networks. The framework enhances the fuel efficiency and throughput of gap-following CACC using a multi-agent deep reinforcement learning bias. Malicious vehicles are identified, their fuel efficiency is reduced through the supply of a dummy bias, and the attacker identities are securely disclosed to other RSUs through a blockchain framework.
2) A method for cooperative verification of Basic Safety Messages (BSMs) which exploits the EIF mechanism to distinguish between legitimate and falsified transmissions.
3) A hybrid consensus protocol merging Zyzzyva Speculative Fault Tolerance with a Joint-Proof-of-Work-Proof-of-Stake consensus mechanism. This protocol is designed to achieve higher throughput and low-latency fault tolerance through the use of speculative techniques and the re-use of information on historical node behaviour.

The rest of the paper is organized as follows. Section II describes the literature survey of the works from which we incorporated elements to develop the Block-MADRL framework. Section III outlines the model of unsignalized intersection. Section IV describes the CACC Control Structure. Section V describes the components of the proposed Block-MADRL framework in detail. Section VI details the simulation methods and performance of Block-MADRL in a real-time arterial network. Section VII concludes the work.

II. Related Works

Recent research shows growing interest in the use of CACC for signalized and unsignalized intersection control [13]. Advances in CACC have mainly focussed on handling communication loss [14], maintaining string stability in mixed traffic [16], and multi-model control [17], [30]. A growing approach to resolving the optimization complexities in vehicular networking is to use a Reinforcement Learning approach [11], [15]. The authors of [18] proposed a fully decentralized and scalable Multi-Agent Reinforcement Learning algorithm for a state-of-the-art advantage actor critic (A2C) agent within the context of Adaptive Traffic Signal Control (ATSC). For the case of unsignalized intersections, the authors of [19] proposed a Co-MADDPG algorithm in which the constraint of a stationary number of agents was overcome by vehicles dynamically selecting several reference vehicles to construct a partial stationary environment. However, these works treat all vehicles as legitimate by assumption. Consequently, untrusted vehicles are rewarded with equal performance as trusted vehicles. In our proposed work, we overcome this by ensuring that untrusted vehicles are denied access to the same learned biases as legitimate vehicles, and we thereby lower the fuel efficiency of the attackers.

The use of blockchain as a decentralized mechanism of handling secure and sensitive information in vehicular networks is becoming increasingly popular in recent literature. In [9], authors use blockchain to achieve consensus among distributed controllers in Software-Defined Vehicular Networks. A duelling Deep Q-Learning approach is used to optimize blockchain throughput. The authors also make use of the Redundant Byzantine Fault Tolerance protocol during consensus. By contrast, the Zyzzyva Speculative Fault Tolerance protocol [27] does not require optimization and is designed to have minimal latency by exploiting speculative mechanisms. As a result, it is highly suitable for the RSU-level blockchain that stores data on untrusted attacker vehicles participating in CACC.

CACC is majorly vulnerable to attacks on the wireless communication channel [4], [20], [21], forgery of acceleration values in BSMs sent from a preceding vehicle to its successor in the string, and attacks on a vehicle's lidar sensors [5]. As solutions to acceleration forgery and lidar attacks, the authors of [5] proposed kinematic verification and a HMM solution to handle the hybrid attacks. We improved the HMM mechanism's efficiency by using the low complexity Isolation Forest and EIF [22] techniques. Apart from reducing complexity, the use of EIF algorithm achieved over 8% improvement in detection rate of subtle attacks when compared to HMM.

III. Unsignalized Intersection Model

In this section, we describe the various entities present in our model of the unsignalized intersection. In this context,

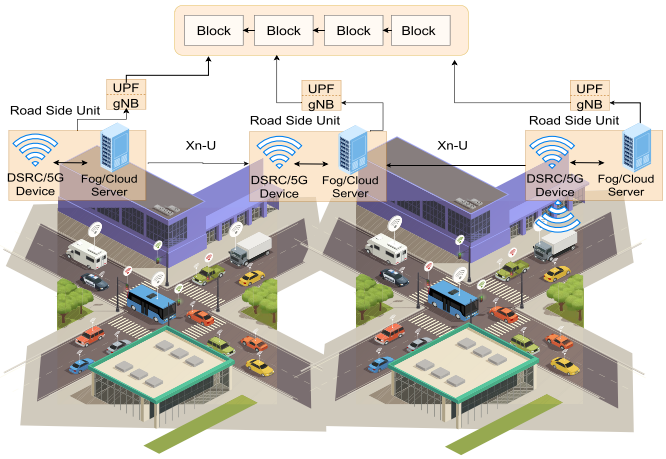

Fig. 1. Unsignalized intersection system.

a road intersection is defined as a point where three or more road segments meet; every road segment is either a one-way or two-way street divided into several lanes. Fig. 1 depicts an example of an unsignalized intersection. An RSU serves as an intersection scheduling center, present in the range of the lanes. The intersection is divided into collision and buffer areas, where the collision area is the collision-prone central region, and the buffer region is the location of vehicle queuing. Within each buffer area of the intersection, there is a set of i input lanes $i \in I = \{1 \dots n\}$. Each input lane has a unique path P_i associated with it. The distance travelled by vehicle i over time t can therefore be written for the path as:

$$d_i(t) = \int_0^t |P_i'(t)| dt \tag{1}$$

where $P_i'(t)$ is the first order derivative of $P_i(t)$ with respect to time t. With this function, we now elucidate on the functioning of unsignalized intersection control.

A. CACC-Enabled Vehicles

Consider two CACC vehicles, V_a and V_b driving on lanes a and b, respectively. The position of V_A on path P_a is such that its distance travelled is $d_a(t_a)$, and likewise, V_b has travelled $d_b(t_b)$. Let the point of intersection of the paths of V_a and V_b be $X_{a,b}$. The distance that V_a would have to travel from origin to $X_{a,b}$ is given by $D_{a,b}$ and likewise, for V_b this distance is $D_{b,a}$. In this scenario, the virtual inter-vehicle distance between V_A and V_B is:

$$\Delta d_{a,b}(t_a, t_b) = d_a(t_a) - d_b(t_b) - L_a - D_{a,b} + D_{b,a} \tag{2}$$

where L_a is the length of vehicle V_a. Vehicles use CACC controllers to set their acceleration rate such that this virtual inter-vehicle distance is close to a pre-defined reference value.

B. Road Side Units

When CACC vehicles arrive within the communication range of the RSU, a communication device collects data on the vehicle's velocity, position and fuel consumption at that instance. This data is passed to a computing server (which may be implemented as either a fog node or a local server) via IPV4/IPV6 backhaul links [24]. As per the 3GPP 5G Architecture [25], RSUs are likely to be modeled as Next-Generation Node Bs (gNBs) when implemented. gNBs communicate with each other via the Xn-U interface, which provides non guaranteed delivery. The standard CACC architecture is sufficient to implement the model. GPUs in RSUs and in-vehicle OBUs enable DRL to be schedulable in standard automotive hardware. Therefore, a low latency, fault tolerant consensus protocol must be used when designing a blockchain operating on the RSU plane. Such a protocol is designed as part of Block-MADRL.

C. Manual Vehicles

To model the dynamics of manually driven, non-CACC vehicles, we assume that they follow the well-validated Intelligent Driver Model (IDM). As per this model, let the position of a vehicle i be x_i at time t, let its velocity be v_i and length be l_i. Let s_i represent the net distance from vehicle i to the vehicle ahead $i-1$. Then:

$$\begin{aligned} s_i &= x_{i-1} - x_i - l_{i-1} \\ \Delta v_i &= v_i - v_{i-1} \end{aligned} \tag{3}$$

The following ordinary differential equations can then describe the velocity of the manual vehicle:

$$x_i = \frac{dx_i}{dt} = v_i \tag{4}$$

$$x_i = \frac{dx_i}{dt} = a\left(1 - \left(\frac{v_i}{v_0}\right)^{\beta} - \left(\frac{s^*(v_i, \Delta v_i)}{s_i}\right)^2\right) \tag{5}$$

where:

$$s^*(v_i, \Delta v_i) = s_0 + v_i T + \frac{v_i \Delta v_i}{2\sqrt{ab}} \tag{6}$$

where v_0 is the desired velocity for the manual vehicle, s_0 is the minimum separation to the ahead vehicle for motion to be possible, T is the desired time headway, a is maximal acceleration and b is comfortable braking deceleration. We set $\beta = 4$ as per standard practice.

IV. CACC Control Structure

In this section, we derive the longitudinal dynamics of vehicles in a platoon. We then propose an enhanced CACC control structure called MADDPG-CACC, which is based on the MADDPG algorithm. In this system, each vehicle uses a lidar sensor to measure its distance to the preceding vehicle. The vehicle receives acceleration information of the preceding vehicle from BSMs, which are broadcast once every 100 ms. The wireless channel operates at a spectral frequency of 75 MHz and uses the 5.9 GHz band [4].

A. Longitudinal Vehicle Dynamics

The common linearized third-order state space representation used for modeling longitudinal vehicle dynamics

is as follows [4]:

$$\frac{\partial q_i}{\partial t} = v_i(t) \tag{7}$$

$$\frac{\partial v_i}{\partial t} = a_i(t) \tag{8}$$

$$\frac{\partial a_i}{\partial t} = \frac{-a_i(t) + u_i(t)}{\eta} \tag{9}$$

where $q_i(t)$, $v_i(t)$, $a_i(t)$, are absolute position, velocity and acceleration of the ith vehicle, and η, $u_i(t)$ represent the internal actuator constant and the commanded acceleration, respectively. The transfer function of the system, $G_i(s)$ can be written as:

$$G_i(s) = \frac{Q_i(s)}{U_i(s)}$$

where $Q_i(s)$ is the Laplace transform for the vehicle position with time, and $U_i(s)$ the Laplace transform of the commanded acceleration for vehicle i. By the properties of Laplace transform, from (7) and (8), we can conclude that:

$$Q_i(s) = \frac{1}{s^2} L(a_i(t))$$

From (9), we can say that:

$$sL(a_i(t)) = \frac{-1}{\eta} L(a_i(t)) + \frac{1}{\eta} U_i(s)$$

Hence on rearranging:

$$U_i(s) = \eta s L(a_i(t)) + L(ai(t)) = (\eta s + 1) L(a_i(t))$$

So:

$$G_i(s) = \frac{Q_i(s)}{U_i(s)} = \frac{1}{s^2(\eta s + 1)} \tag{10}$$

B. MADDPG-CACC Control Structure

We first introduce the control structure for Adaptive Cruise Control (ACC) and show how this is modified to produce constant time gap CACC. CACC is then modified to produce our proposed MADDPG-CACC control structure. Firstly, a signal conditioning block called $H_i(s)$ is utilized for implementing a constant time gap (headway) policy. It is given by:

$$H_i(s) = 1 + h_{d,i} s \tag{11}$$

where $h_{d,i}s$ is the constant time headway. The next controller, $C_{i,ACC}(s)$ is a feedback Proportional-and-Derivative (PD) controller which uses data from the lidar to sense distance to the vehicle ahead. Taking input as the error between the existing and desired positions, this controller outputs the desired acceleration. The control equation in the Laplace domain is:

$$U_{fb,i}(s) = C_{i,ACC}(s) E_i(s) = \omega_{c,i}(\omega_{c,i} + s) E_i(s) \tag{12}$$

In this scenario, $\omega_{c,i}$ is the bandwidth of the controller, chosen such that $\omega_{c,i} = 1/\eta$ holds, in order to prevent actuator saturation. $E_i(s)$ is the Laplace transform of spacing error. The spacing error is:

$$e_i = q_{i-1} - q_i - h_{d,i} v_i - r_i \tag{13}$$

where r_i forms the gap between consecutive vehicles at standstill. In this scenario, the time-domain equivalent of (12) is obtained by using (13) such that:

$$u_{fb,i} = \omega_{c,i}^2 + \omega_{c,i} \frac{\partial e}{\partial t}$$

$$u_{fb,i} = \omega_{c,i}^2 + \omega_{c,i}(v_{i-1} - v_i - h_{d,i} a_i) \tag{14}$$

where $\omega_{c,i}^2$ and $\omega_{c,i}$ represent the proportional and derivative gains of the ACC controller, respectively. Equation (14) shows that the feedback ACC controller relies on sensor measurements of relative velocity and local acceleration measurements of the ego vehicle to produce the commanded acceleration. In order to improve vehicle following and string stability, the ACC controller is compounded by a CACC part, for which the ego vehicle requires information on the acceleration of the preceding vehicle. The transfer equation for this CACC controller can be written as:

$$U_{ff,i}(s) = C_{i,CACC}(s) A_{i-1}(s) = \frac{1}{H_i(s) G_i(s) s^2} A_{i-1}(s) \tag{15}$$

where $A_{i-1}(s)$ is the Laplace transform of the acceleration for the vehicle with index $i-1$. This feed-forward acceleration designed to achieve zero steady state vehicle following error. Now, using the fact that $A_{i-1}(s) = s^2 Q_{i-1}(s)$, and that from (10), we know that $Q_{i-1}(s) = G_{i-1}(s) U_{i-1}(s)$, we can rewrite (15) as:

$$U_{ff,i}(s) = \frac{1}{H_i(s) G_i(s)} G_{i-1}(s) U_{i-1}(s) \tag{16}$$

Because all the vehicles in the string are homogeneous, they have constant gains, and so $G_{i-1}(s) = G_i(s)$. Equation (16) thus reduces to:

$$U_{ff,i}(s) = \frac{U_{i-1}(s)}{H_i(s)} \tag{17}$$

Due to the serial combination of $U_{ff,i}$ and $U_{fb,i}$ to get $U_i(s)$ we know that:

$$U_i(s) = U_{ff,i}(s) + U_{fb,i}(s) \tag{18}$$

Thus on substituting (17) and (12) into (18), we receive:

$$\begin{aligned} U_i(s) &= \frac{U_{i-1}(s)}{H_i(s)} + \omega_{c,i}(\omega_{c,i} + s) E_i(s) \\ &= \frac{U_{i-1}(s)}{H_i(s)} + {\omega_{c,i}}^2 E_i(s) + \omega_{c,i} s E_i(s) \end{aligned} \tag{19}$$

In the time domain, (19) can be written as:

$$\begin{aligned} u_i(t) = \frac{1}{h_i(t)}(u_{i-1}(t)) + {\omega_{c,i}}^2 (q_{i-1} - q_i - h_{d,i} v_i - r_i) \\ + \omega_{c,i}(v_{i-1} - v_i - h_{d,i} a_i) \end{aligned} \tag{20}$$

This equation can be captured in a more practical format by the gap following equation provided in [26]. This states that the desired acceleration for a vehicle in gap following mode is given by:

$$a_g^* = K_a a_p + K_v(v_p - v) + K_g(g - G_{min} - v T_g) \tag{21}$$

where a_g^* is the desired acceleration of the ego vehicle, a_p is the acceleration of the preceding vehicle, v_p is the velocity of the preceding vehicle, v is the velocity of the ego vehicle, K_a is the acceleration gain of the preceding vehicle, K_v is speed gain, g is the inter-vehicle distance, T_g is headway time constant, K_g is gap gain, and G_{min} is the minimum space gap between two subsequent vehicles.

The control structure for MADDPG-CACC adds a learned value $F(a_p, g)$ to the desired acceleration equation in (21). The function F is learned through multi-agent deep reinforcement learning. $F(a_p, g)$ aims to compensate for errors in the stated value of a_p, and to compensate for errors in estimating parameters K_a, K_v, K_g. $F(a_p, g)$ serves to optimize the vehicle's fuel consumption and maximize the achieved throughput of the lane. The bias also learns to spot patterns of falsified preceding acceleration and tunes the desired acceleration to compensate for fraudulent messages, thus providing better performance compared to CACC in hostile environments. Lastly, the bias is used to reduce the fuel efficiency of identified malicious attackers selectively. The method for MADDPG-CACC is summarized in Algorithm 1.

Algorithm 1 MADDPG-CACC

Input : Number of episodes (M), maximum episode length ($max - ep$), Local Blacklist (LB), Set of Agents (Ag), Trust Blockchain (B), Experience Replay Buffer (E), Dummy policy for blacklisted agents (μ^{Bl}), Critic Parameter (θ)
Output: Set of Agent Actions ($a_1, a_2..a_N$)

for *episode = 1 to M* **do**
  Initialize a random process for action exploration
  Receive initial state X of system
  Initialize empty history h for all agents
  for $t = 1$ *to* $max - ep$ **do**
    for *each agent i available during episode t* **do**
      Get current state(x_i)
      if $i \in LB$ **then**
        $a_i = \mu_{\theta_i}^{Bl}(x_i, h_i)$
      end
      else
        $a_i = \mu_{\theta_i}(x_i, h_i)$
      end
      Set desired acceleration of vehicle i by (22)
      where $F(a_p, g) = a_i$
      Execute action a_i, obtain reward r_i from (23),
      Obtain new state x_i'
      Send (x_i, a_i, r_i, x_i') in update request to E
    end
    for $i \notin LB$ **do**
      Sample a mini-batch of S samples (x^j, a^j, r^j, h'^j) from E
      Set y^j as: $r_i^j + \gamma * Q_i^{\mu'}(x^{j'}, h^{j'}, a_1', a_2'..a_N')|_{a_k'=\mu_k'(s_k^j, h_k^j)}$
      Update the critic network by minimizing loss:
      $\mathcal{L}(\theta_i) = \frac{i}{S}(\sum_j y^i - Q_i^{\mu'}(x^j, h^j, a_1, ..a_N))^2$
      Update actor network using $\nabla_{\theta_i} J$ value:
      $\frac{i}{S}\sum_j \nabla_{\theta_i}\mu_i(s_j^i, h_j^i)\nabla_{a_i} Q_i^{\mu}(x^j, h^j, a_1, ..a_N)$
      $\theta_i' \leftarrow \tau\theta_i + (1-\tau)\theta_i'$
    end
  end
end

V. Block-MADRL Framework

This section describes the multi-agent deep reinforcement learning, attack detection and blockchain consensus components of Block-MADRL.

A. Multi-Agent Deep Reinforcement Learning

The target of the multi-agent deep reinforcement learning component of Block-MADRL is to learn a function $F(a_p, g)$. This function calculates a bias value which is added to the standard CACC desired acceleration. The standard desired acceleration for CACC is given by (21), while the proposed MADDPG-CACC control structure uses a modified acceleration given by:

$$a_g^{**} = K_a a_p + K_v(v_p - v) + K_g(g - G_{min} - vT_g) + F(a_p, g) \tag{22}$$

The function F is learned such that the target vehicle will experience optimal fuel efficiency and lane throughput when using desired acceleration a_g^{**} rather an a_g^*.

1) State Space and Action Space Specification: The state of a vehicle x in a set of vehicles V is given by a vector $S_x = [a_p, \Delta g(t)]$ where a_p is the acceleration of the directly preceding vehicle in the platoon and '$\Delta g(t)$' is given by:

$$\Delta g(t) = |g(t) - g(t - T)|$$

where $g(t)$ refers to the distance between the target and preceding vehicles at time-step t, and T refers to the time period between subsequent measurements of inter-vehicle distance. The value of $g(t)$ for a leading vehicle in the platoon or a vehicle which is not part of a platoon is made zero, indicating that the vehicle is not in gap following mode. In this case, the MADDPG-CACC algorithm does not apply to the vehicle, and the vehicle's fuel consumption and throughput during that state are not considered during performance evaluation. The action space of the vehicle consists of a single value $F(a_p, g)$ indicating the learned bias to be added to the desired acceleration of standard gap following CACC.

2) Reward Design: Biases that lead to greater fuel efficiency, higher throughput and lower collisions are to be positively rewarded. In this respect, collisions (or violations of the minimal inter-vehicle distance $G_{min} = 2m$) must be avoided with utmost importance. Hence we design a reward function given by:

$$R(S_v) = -\alpha * num_{collisions} - \beta * fuel + \beta * \theta \tag{23}$$

The parameter $num_{collisions}$ is the number of collisions undergone by the vehicle in state S_v, $fuel$ is fuel consumption in terms of litres consumed per 100km travelled (L/100km) and θ is the throughput in terms of vehicles per hour (vph) passing through the intersection. We assume homogeneous vehicle length L. The throughput (θ) of a vehicle V_a travelling at velocity v_a and whose gap from the preceding vehicle is given by g is [13]:

$$\theta = \frac{v_a}{g + L}$$

3) Multi-Agent Deep Deterministic Policy Gradient: The unsignalized intersection is modelled as a multi-agent environment. N_t vehicles present in the range of the RSU at time t are expressed as Markov Decision Making Process (MDP) agents. Every time a new vehicle enters the communication

Algorithm 2 Attack Detection and Mitigation

Input : Basic Safety Message (BSM), Time Window (T), Receiver Vehicle (V_r), Transmitter Vehicle (V_t), Anomaly Threshold (θ)
Output: Local Blacklist (LB)

```
//At V_r
Send BSM to neighbour
Send sensor estimate of V_t velocity, V_t position to neighbour
for each neighbour in LOS of V_r do
    if VehicleMode=="Verifier" then
        Calculate vector D[V_t] of differences between sensor estimate of V_t velocity, position with V_r's sensor estimate of V_t velocity, position
        Add difference between reported acceleration in BSM and kinematic estimate of V_t acceleration to D[V_t]
        p = CreatePacket(V_t, V_r, D[V_t])
        Send packet p to nearest RSU
    end
end
//At RSU
if more than 50% verifiers supply D[V_t] that is anomalous as per θ then
    V_t is added to LB
end
else
    V_t stays trusted or is removed from LB
end
Send LB-V to all V ∈ LB, send LB to all V ∉ LB
//At V_r
if V_t ∈ LB then
    Switch to sensor-based calculation of acceleration
    Inform platoon leader and follower vehicles about formation
    of new platoon with V_r as leader
    while V_t stays in front of V_r do
        Continue in sensor-calculation mode
    end
    Request merging with platoon ahead
    Switch back to original CACC mode
end
return LB
```

Algorithm 3 Zyzzyva-Joint-Proof-of-Work-Proof-of-Stake Consensus Algorithm

Input : Current Blockchain (B), Set of RSUs with update request (R), Block Reward (BR), Failure tolerance level (f)
Output: New Blockchain (B')

```
for rsu ∈ R do
    Stake[rsu] = Balance of rsu from successful mining instances
    Updates[rsu] = Number of changes to last committed local blacklist //where f(x) is a monotonically increasing function
    Threshold[rsu] = f(Stake[rsu]+Updates[rsu])
    Calculate Nonce[rsu]
end
Miner RSU ← rsu ∈ R such that (Threshold[rsu]-Nonce[rsu]) is maximum
Miner RSU sends its candidate block to pre-elected Primary RSU
Primary RSU sends candidate block in an < OrderRequest > operation to other RSUs
Non-primary RSUs speculatively add the latest blacklisted vehicle IDs in candidate block to their local blacklist
Non-primary RSUs verify the hashes of sequence number and candidate block data
Non-primary RSUs notify their acceptance/rejection of block to the primary RSU
c = Number of RSUs accepting candidate block
if c == 3f + 1 then
    Miner's blockchain (B') accepted
end
if 2f + 1 ≤ c ≤ 3f then
    Activate Two-Phase Case of Zyzzyva Fault Tolerance Protocol [27]
end
else
    Activate 3a and 3b cases of Zyzzyva Fault Tolerance Protocol [27]
end
Resolve Consensus Blockchain B'
Add BR to Miner RSU Account
return B'
```

range of its nearest RSU, the RSU initializes a corresponding agent with the currently available centralized actor and critic networks. Let the state S represent each of the N_t agents' collective states and let their collective Action A be $(a_1, a_2 \ldots a_n)$. On execution, the agent i receives a reward based on fuel consumed and current throughput. All agents are assumed to share a common reward function (r_i) and discount factor (γ). The critic receives the following target output:

$$y^i = r_i^j + \gamma * Q_i^{\mu'}(x^{j'}, h^{j'}, a_1', a_2'..a_N')|_{a_k'=\mu_k'(s_k^j, h_k^j)} \quad (24)$$

where y^i is the Q value obtained according to policy μ_i for agent i under state s_i and with mini-batch size S. The critic modifies its parameters to maximize reward and minimize loss function $L(\theta_i)$:

$$\mathcal{L}(\theta_i) \approx \frac{i}{S}(\sum_j y^i - Q_i^{\mu'}(x^j, h^j, a_1, ..a_N))^2 \quad (25)$$

Optimal parameters for the actor network are then found by performing Stochastic Gradient Ascent (SGA) update as:

$$\nabla_{\theta_i} J = \frac{i}{S}\sum_j \nabla_{\theta_i}\mu_i(s_j^i, h_j^i)\nabla_{a_i} Q_i^{\mu}(x^j, h^j, a_1, ..a_N) \quad (26)$$

To cope with the non-stationary number of agents, we implement the MADDPG agent as a single collective DDPG agent, whose common experience replay buffer is updated by the various vehicles in response to their individual states and rewards. However, while all legitimate vehicles access the commonly learnt policy, blacklisted vehicles will only be provided actions learned from a dummy policy. Hence vehicular agents are distinguished by their status of legitimacy.

B. Attack Detection Model

The attack detection strategy is explained in Algorithm 2. As per the algorithm, every trusted vehicle must periodically verify BSMs received within range through sensor based estimation. The verifying vehicle constructs a vector of differences D, consisting of the differences between the verifying vehicle and receiver vehicle's sensor estimates of the transmitter vehicle's velocity and position. The vector also includes the difference between the kinematic estimate and the reported value of the transmitter vehicle's acceleration. The overall vector D is sent to the RSU, where it is subsequently passed through an EIF. The EIF is trained on a ground truth of difference vectors for a non-attack scenario. The ground truth data set was generated from the OMNET++ simulator environment being run on a scenario where no attack was prevalent.

The same environment was used in simulating the attack. This was to ensure appropriate control of variables and validity of the calculated impact of the attack. If most of the difference vectors supplied by various verifiers for a given transmitter vehicle are found anomalous, the transmitter vehicle is blacklisted by the RSU. Otherwise, the vehicle is removed from blacklist or simply not added. The RSU sends a copy of its

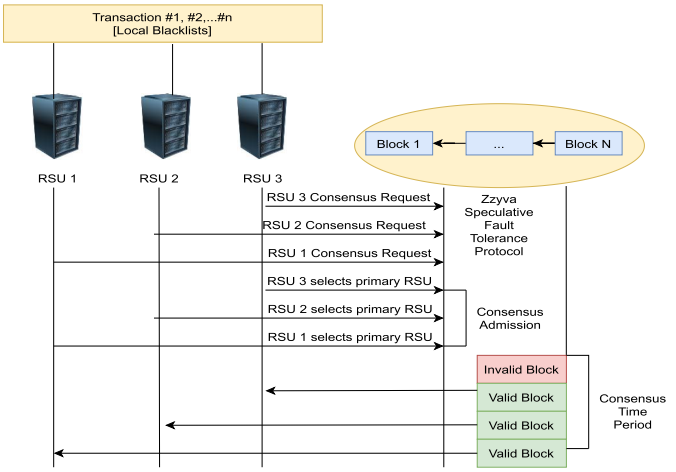

Fig. 2. RSU-Blockchain interaction.

local blacklist to all vehicles, who use it to filter their received BSMs. Blacklisted vehicles receive information on all other blacklisted vehicles except themselves. This is to prevent the blacklisted vehicles from being aware that they have been detected. If aware of being detected, the blacklisted vehicle could change its attack method or stop using the dummy biases it receives from the RSU. The RSU then attempts to mine its local blacklist to the blockchain to notify other RSUs of the malicious vehicles. Similarly, the latest malicious vehicles present in the blockchain are added to the local blacklist of the RSU.

C. RSU Level Blockchain

We propose a hybrid Zyzzyva-Joint-Proof-of-Work-Proof-of-Stake consensus smart contract for the RSU-level blockchain. In this system, a Joint-Proof-of-Work-Proof-of-Stake miner election method is designed, which takes the historical profits of the RSUs as stake, and the difficulty to complete the Proof-of-Work depends on the stake. RSUs with higher stakes solve the nonce value and win the miner election more easily, which ensures the timely update of data stored in the blockchain. Post miner election, the other RSUs verify the proposed block's authenticity and reach consensus through the Zyzzyva Speculative Fault Tolerance protocol. Further details are specified in Algorithm 3. Fig. 2 captures the interaction between the RSUs and the blockchain.

VI. Performance Evaluation

The Block-MADRL framework is evaluated in a real world double-intersection traffic network of Downtown Orlando, extracted using OpenStreetMap. The mobility environment is provided by SUMO. We use the OMNET++ discrete event simulator framework, integrated with VEINS and INET to achieve inter-vehicular communication. The vehicles interface with 10 terminal servers that simulate real-time RSUs. The BlockSim framework [29] is extended to implement Zzyva Speculative Fault Tolerance and Joint-Proof-of-Work-Proof-of-Stake consensus algorithms, using the RSU servers as blockchain nodes. The deep reinforcement learning algorithms are implemented in the RSU server through PyTorch library functions. Simulation parameters are listed in Table I, while deep learning parameters are specified in Table II.

TABLE I
Simulation Parameters

Parameter	Value
Number of vehicles	30
Simulation Time (s)	60
Frequency Band (GHz)	5.9
Communication Protocol	DSRC
Time period between BSMs (s)	0.1
Number of RSUs	10
Radio Transmitter Power (mW)	20
Radio Bandwidth (MHz)	10
Intersection Area (m^2)	1.254×10^4
Communication Region(m)	500

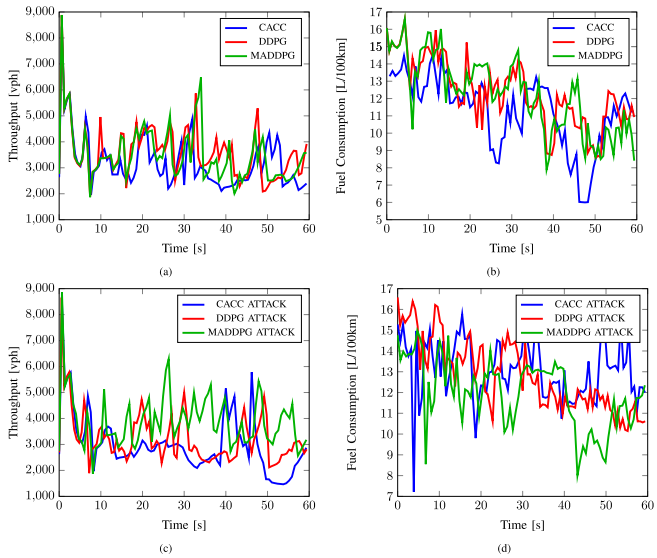

Fig. 3. Comparison of CACC Techniques (a) No Attack Throughput (b) No Attack Fuel Consumption (c) Attack Throughput (d) Attack Fuel Consumption.

TABLE II
Deep Reinforcement Learning Parameters

Parameter	Value
Hidden Layers	2
Neurons per hidden layer	64
Learning Rate	0.01
Time steps per episode	40
Size of mini-batch	78
γ	0.99
Optimizer	Adam

A. Performance of MADDPG-CACC

We evaluated three different CACC techniques - standard gap following (CACC) [25], CACC with biases learnt from individual observation via DDPG (DDPG), and the proposed MADDPG-CACC control structure of Block-MADRL (MADDPG). These techniques are tested for a period of sixty seconds [14] under a normal environment, followed by an environment where 50% of the vehicles transmit false acceleration as: $a_{fake} = a_{real} + c_a sin(ft)$. The value a_{fake} is the

TABLE III
COMPARISON OF CACC TECHNIQUES

Scenario	Through-put(vph)	Fuel (L/100km)	Reward (rew/s)
Normal CACC	3180	12.42	-47.4
DDPG-based CACC	**3593.6**	12.35	-44.21
MADDPG-CACC	3507.2	**12.24**	**-43.99**
Normal CACC (Attack)	2995.2	**11.317**	-44.404
DDPG-based CACC (Attack)	3225.6	12.82	-48.155
MADDG-CACC (Attack)	**3881.6**	11.967	**-42.378**

TABLE IV
LEGITIMATE VS. BLACKLISTED VEHICLE PERFORMANCE

Attacker	Defence	Vehicle	Through-put (vph)	Fuel (L/100km)	Reward (rew/s)
Even	Block-MADRL	Legitimate	**3616**	**9.69**	**-30.511**
		Blacklisted	3456	11.96	-45.728
Odd	Block-MADRL	Legitimate	**1196.4**	**11.52**	**-42.562**
		Blacklisted	3430.4	13.09	-50.096
	CACC Defence	Legitimate	3318.4	10.204	-38.652
		Blacklisted	**2470.4**	**8.52**	**-34.25**

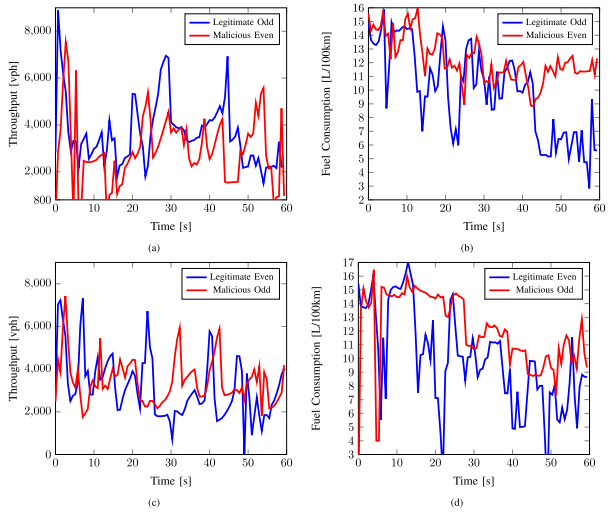

Fig. 4. Block-MADRL impact on blacklisted vehicles (a) Even attack throughput (b) Even attack fuel (c) Odd attack throughput (d) Odd attack fuel.

false transmitted acceleration, a_{real} is the actual acceleration of the attacker, and $c_a = 5$, $f = 5$ (in accordance with the authors of [5]).

The average throughput and fuel consumed per second by each vehicle are captured in Table III, and graphically represented in Fig. 3. As expected, MADDPG-CACC maximizes rewards in both scenarios; it achieves 10.2% more throughput than standard CACC for a similar fuel consumption rate. Because the optimal policy is relatively simpler to learn in a non-attack scenario, the DDPG-based CACC also performs relatively well. However, in the scenario of attack, it is seen that individual observations are insufficient for learning and thus DDPG-based CACC performs worse than standard CACC. By contrast, the MADDPG-CACC achieves 29.5% more flow than standard CACC and 20% more flow than DDPG-based CACC, with a similar fuel consumption rate.

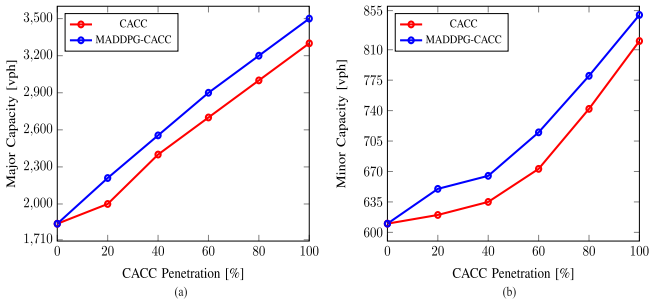

Fig. 5. Intersection capacity with market penetration rate of CACC at (a) Major intersection (b) Minor intersection.

Another advantage of MADDPG-CACC is that it allows RSUs to selectively reduce the fuel efficiency of attackers.

B. Performance Analysis of Block-MADRL Defence Mechanism

As part of the Block-MADRL defence mechanism, we do not provide MADDPG-learned bias values to vehicles in the local blacklist. Instead, we feed the bias values of a dummy policy to the blacklisted vehicle, and refrain from notifying the vehicle that its malicious behaviour has been identified. The idea is to punish illegitimate behaviour by withdrawing access to learned biases, and induce lower fuel efficiency in malicious vehicles. This will contribute to increased cost of attack and the speedy draining of attacker resources. Non-blacklisted vehicles receive valid, learned biases and are not affected.

We tested performance during two acceleration attacks to compare the overall effect of blacklisting on the rewards available to malicious vehicles. In the first, all vehicles with even identities transmitted false accelerations, and in the second, vehicles with odd identities performed the same attack. We compared the effect of supplying dummy bias values to blacklisted vehicles while using Block-MADRL with the regular defence mechanism of using sensory estimates of acceleration with standard CACC. The results are summarized in Table IV and plotted in Fig. 4. Rewards are measured in reward per second (rew/s). The use of the dummy bias value mechanism achieved over 23% decrease in the fuel efficiency of malicious even identity vehicles, and 13.6% decrease in fuel efficiency of malicious odd identity vehicles compared to their legitimate counterparts.

C. Block-MADRL Impact on Intersection Capacity With Varying Market Penetration Rates

We conducted experiments on two alternate intersections in Downtown Orlando with minor and major capacities. We incrementally loaded traffic into the network and measured the average intersection throughput every fifteen minutes to calculate capacity. Traffic was added until the throughput reached a steady state and no longer rose with demand. This maximal throughput was used as intersection capacity [23]. Fig. 5(a) depicts the minor intersection capacity and 5(b) the major intersection capacity with various rates of CACC market penetration. Vehicles that do not use MADDPG-CACC or CACC follow the IDM driver model.

TABLE V
ATTACK SCENARIOS VS. DETECTION RATES IN VARIOUS SCHEMES

	Kinematics Detection Rate (%)	Isolation Forest Detection Rate (%)	Extended Isolation Forest Detection Rate (%)	PHY Detection Rate [5] (%)	HMM Detection Rate [5] (%)
No Attack (False Positives)	15.3	16.4	13.3	**0.35**	1.5
ACL (c_a = 5, f = 5)	75.22	**88.34**	87.77	25.75	77.5
VEL (c_v = 1)	89.76	88.50	90.07	**95.13**	83.5
POS (c_d = 0.1)	89.76	87.42	**90.07**	0.25	74
VEL-POS (c_v = 0.2, c_d = 0.1)	89.76	87.42	**90.07**	0.13	90
Overall Average Detection Rate	88.94	87.92	**89.495**	30.15	81.25

TABLE VI
CONSENSUS PROTOCOL PERFORMANCE

Protocol	Throughput (tx/s)	Average Profit per RSU (BTC)
Zyzzyva-Fork	166.6985	5379.18
Longest Fork	192.9428	6235.585
Zyzzyva-Joint-PoW-PoS	196.0844	6337.877
Joint-PoW-PoS	**209.877**	**6787.979**

It is observed that intersection capacity increases by 90.2% in the major intersection and 39.3% in the minor intersection when MADDPG-CACC market penetration is at 100%. Comparatively, the use of CACC increases intersection capacity by 79.3% in the minor and 34.4% in the major intersection at a penetration rate of 100%. The linear trend in the major capacity intersection can be attributed to the increased influence of lane changing behavior on capacity with CACC penetration.

By contrast, the minor capacity lanes are too small for lane changes to occur reliably, so these happen upstream of the intersection. The diminished effect of lane changes on capacity leads to the parabolic minor capacity curve. The increased capacity afforded by MADDPG-CACC is attributed to the learned bias, which seeks to maximize lane throughput and therefore adapts CACC from a linear formula (which may not estimate all parameters accurately) to one optimized to the specific intersection environment.

D. Performance of Attack Detection

To find the efficiency of the attack detection model, we compare the detection rates with those obtained from a kinematics-based detection mechanism and on the HMM based detection method. We tested the following anomaly detection approaches: kinematic verification using three checks (as opposed to the PHY approach in [5], which only uses two), Isolation Forest mechanism and the EIF mechanism. The attacks that were tested upon were [5]:

- Acceleration Attack (ACL): $a_{fake} = a_{real} + c_a sin(ft)$
- Velocity Attack (VEL): $v_{fake} = v_{real} - c_v t$
- Position Attack (POS): $d_{fake} = d_{real} + c_d t$
- Hybrid velocity-position attack (VEL-POS)

where v_{fake} and d_{fake} are the false speed and distance values perceived by vehicles under lidar attack, and v_{real}, d_{real} are the actual speed and distance values. Co-efficients c_a, c_v and c_d are set to the values in [5] for ease of comparison. Table V summarizes detection rate results. It is found that the EIF method outperforms HMM by over 8% in terms of overall detection rate, whilst reducing algorithmic complexity from $O(n^2)$ to $O(n)$. EIF is thus the optimal approach for attack detection.

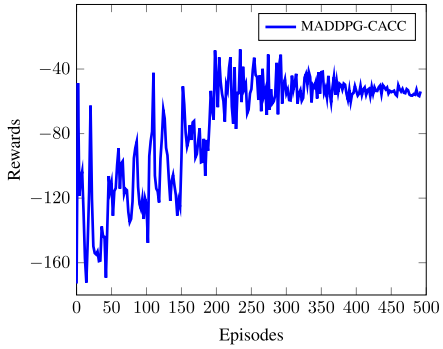

Fig. 6. Training graph for MADDPG-CACC.

E. Performance of Blockchain Consensus Protocol

We analyzed the relative performance of the blockchain across four alternative blockchain consensus protocols: Longest Fork (standard benchmark), Joint-Proof-of-Work-Proof-of-Stake, as well as two hybrid consensus mechanisms involving Zyzzyva Speculative Fault Tolerance: Zyzzyva-Fork and Zyzzyva-Joint-Proof-of-Work-Proof-of-Stake (Zyzzyva-Joint-PoW-PoS). The profit and throughput of the blockchain under the various consensus protocols are summarized in Table VI.

We found that the Joint-Proof-of-Work-Proof-of-Stake based consensus which we developed provided 8.08% higher throughput compared to regular longest fork consensus.

The use of Zyzzyva Speculative Fault Tolerance reduces profits and throughput, on account of the extra network delay of signing and verification. However, this protocol provides essential fault tolerance at a cost of only 7% lower throughput compared to regular Joint-Proof-of-Work-Proof-of-Stake. Zyzzyva-Joint-Proof-of-Work-Proof-of-Stake also has 17.6% improvement in throughput and profit compared to Zyzzyva blended with Longest Fork consensus. Thus the best trade-off between efficiency and fault tolerance can be achieved by adopting Zyzzyva-Joint-Proof-of-Work-Proof-of-Stake consensus for the RSU-level blockchain.

F. Convergence of Training

The convergence of the training process is depicted in Fig. 6. We trained the environment with MADDPG-CACC

(under no attack) on an arbitrary intersection in Downtown Orlando (different from that of testing) for a period of over 500 episodes, during which the average overall rewards for each vehicle converted to around -51.7. The percentage of collisions reduced from 30.4% close to the start of training to around 0.2% towards the ending, as collision-prone policies were heavily penalized by the reward function. The model was made to perform online learning in the test environment, whereby new experiences would consistently be updated in the RSU replay buffers and used to control vehicle actions.

VII. Conclusion

In this paper we presented Block-MADRL, an integrated framework for enhancing CACC efficiency, cooperatively identifying malicious insider CACC vehicles, reducing the fuel efficiency of the identified attackers, and securely disseminating attacker data to other RSUs through a low latency, fault tolerant blockchain. The use of MADDPG-CACC in the absence of any defence mechanism achieved 29.5% better throughput compared to standard CACC under attack. On employing Block-MADRL, attackers were identified with an overall 8% higher detection rate compared to state-of-the-art and the fuel efficiency of identified attackers was reduced by up to 23%. Lastly, attacker identity data was securely distributed across RSUs through a low latency, fault-tolerant Zyzzyva-Joint-PoW-PoS protocol, which achieved 17.6% better blockchain throughput compared to Zyzzyva-Fork consensus.

References

[1] G. Raja, S. Anbalagan, G. Vijayaraghavan, P. Dhanasekaran, Y. D. Al-Otaibi, and A. K. Bashir, "Energy-efficient end-to-end security for software-defined vehicular networks," *IEEE Trans. Ind. Informat.*, vol. 17, no. 8, pp. 5730–5737, Aug. 2021, doi: 10.1109/TII.2020.3012166.

[2] B. Qian, H. Zhou, F. Lyu, J. Li, T. Ma, and F. Hou, "Toward collision-free and efficient coordination for automated vehicles at unsignalized intersection," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10408–10420, Dec. 2019.

[3] A. I. Morales Medina, F. Creemers, E. Lefeber, and N. van de Wouw, "Optimal access management for cooperative intersection control," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 5, pp. 2114–2127, May 2020.

[4] A. Alipour-Fanid, M. Dabaghchian, and K. Zeng, "Impact of jamming attacks on vehicular cooperative adaptive cruise control systems," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 12679–12693, Nov. 2020.

[5] M. Jagielski, N. Jones, C.-W. Lin, C. Nita-Rotaru, and S. Shiraishi, "Threat detection for collaborative adaptive cruise control in connected cars," in *Proc. 11th ACM Conf. Secur. Privacy Wireless Mobile Netw.*, Stockholm, Sweden, Jun. 2018, pp. 184–189.

[6] J. Yang, X. Liu, S. Liu, D. Chu, L. Lu, and C. Wu, "Longitudinal tracking control of vehicle platooning using DDPG-based PID," in *Proc. 4th CAA Int. Conf. Veh. Control Intell. (CVCI)*, Hangzhou, China, Dec. 2020, pp. 656–661.

[7] A. Peake, J. McCalmon, B. Raiford, T. Liu, and S. Alqahtani, "Multi-agent reinforcement learning for cooperative adaptive cruise control," in *Proc. IEEE 32nd Int. Conf. Tools Artif. Intell. (ICTAI)*, Baltimore, MD, USA, Nov. 2020, pp. 15–22.

[8] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. NeurIPS*, Long Beach, CA, USA, 2017, pp. 6382–6393.

[9] D. Zhang, F. R. Yu, and R. Yang, "Blockchain-based distributed software-defined vehicular networks: A dueling deep *Q*-learning approach," *IEEE Trans. Cogn. Commun. Netw.*, vol. 5, no. 4, pp. 1086–1100, Dec. 2019.

[10] G. S. Aujla, A. Singh, M. Singh, S. Sharma, N. Kumar, and K.-K.-R. Choo, "BloCkEd: Blockchain-based secure data processing framework in edge envisioned V2X environment," *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 5850–5863, Jun. 2020.

[11] G. Raja, A. Ganapathisubramaniyan, S. Anbalagan, S. B. M. Baskaran, K. Raja, and A. K. Bashir, "Intelligent reward-based data offloading in next-generation vehicular networks," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 3747–3758, May 2020.

[12] P. K. Singh, R. Singh, S. K. Nandi, and S. Nandi, "Integrating blockchain with CACC for trust and platoon management," in *Cryptocurrencies and Blockchain Technology Applications*. Hoboken, NJ, USA: Wiley, 2020, pp. 77–97.

[13] D. Kalathil, A. A. Kurzhanskiy, P. P. Varaiya, and J. Perez, "Sustainable operation of arterial networks," California Dept. Transp., Sacramento, CA, USA, Tech. Rep. CA17-3001, 2017.

[14] C. Wu, Y. Lin, and A. Eskandarian, "Cooperative adaptive cruise control with adaptive Kalman filter subject to temporary communication loss," *IEEE Access*, vol. 7, pp. 93558–93568, 2019.

[15] W. Gao, J. Gao, K. Ozbay, and Z.-P. Jiang, "Reinforcement-learning-based cooperative adaptive cruise control of buses in the Lincoln tunnel corridor with time-varying topology," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 10, pp. 3796–3805, Oct. 2019.

[16] Y. Qin and S. Li, "String stability analysis of mixed CACC vehicular flow with vehicle-to-vehicle communication," *IEEE Access*, vol. 8, pp. 174132–174141, 2020.

[17] F. Navas, V. Milanes, C. Flores, and F. Nashashibi, "Multi-model adaptive control for CACC applications," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 2, pp. 1206–1216, Feb. 2021.

[18] T. Chu, J. Wang, L. Codecà, and Z. Li, "Multi-agent deep reinforcement learning for large-scale traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 1086–1095, Mar. 2020.

[19] T. Wu, M. Jiang, and L. Zhang, "Cooperative multiagent deep deterministic policy gradient (CoMADDPG) for intelligent connected transportation with unsignalized intersection," *Math. Problems Eng.*, vol. 2020, Jul. 2020, Art. no. 1820527.

[20] A. Alipour-Fanid, M. Dabaghchian, H. Zhang, and K. Zeng, "String stability analysis of cooperative adaptive cruise control under jamming attacks," in *Proc. IEEE 18th Int. Symp. High Assurance Syst. Eng. (HASE)*, Singapore, Jan. 2017, pp. 157–162.

[21] A. Alipour-Fanid, M. Dabaghchian, and K. Zeng, "Platoon stability and safety analysis of cooperative adaptive cruise control under wireless rician fading channels and jamming attacks," 2017, *arXiv:1710.08476*.

[22] S. Hariri, M. C. Kind, and R. J. Brunner, "Extended isolation forest," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 4, pp. 1479–1489, Apr. 2021.

[23] H. Liu, X.-Y. Lu, and S. E. Shladover, "Traffic signal control by leveraging cooperative adaptive cruise control (CACC) vehicle platooning capabilities," *Transp. Res. C, Emerg. Technol.*, vol. 104, pp. 390–407, Jul. 2019.

[24] S. Anbalagan *et al.*, "Machine-learning-based efficient and secure RSU placement mechanism for software-defined-IoV," *IEEE Internet Things J.*, vol. 8, no. 18, pp. 13950–13957, Sep. 2021, doi: 10.1109/JIOT.2021.3069642.

[25] *Architecture Enhancements for 5G System (5GS) to Support Vehicle-to-Everything (V2X) Services*, (3GPP TS 23.287 version 16.3.0 Release 16), document ETSI TS 123 287 V16.3.0, Jul. 2020.

[26] B. Tian, X. Deng, Z. Xu, Y. Zhang, and X. Zhao, "Modeling and numerical analysis on communication delay boundary for CACC string stability," *IEEE Access*, vol. 7, pp. 168870–168884, 2019.

[27] R. Kotla, L. Alvisi, M. Dahlin, A. Clement, and E. Wong, "Zyzzyva: Speculative Byzantine fault tolerance," *ACM Trans. Comput. Syst.*, vol. 27, no. 4, pp. 1–39, 2010.

[28] G. Raja, S. Anbalagan, G. Vijayaraghavan, S. Theerthagiri, S. V. Suryanarayan, and X.-W. Wu, "SP-CIDS: Secure and private collaborative IDS for VANETs," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4385–4393, Jul. 2021, doi: 10.1109/TITS.2020.3036071.

[29] M. Alharby and A. van Moorsel, "BlockSim: An extensible simulation tool for blockchain systems," 2020, *arXiv:2004.13438*.

[30] A. Petrillo, A. Pescape, and S. Santini, "A secure adaptive control for cooperative driving of autonomous connected vehicles in the presence of heterogeneous communication delays and cyberattacks," *IEEE Trans. Cybern.*, vol. 51, no. 3, pp. 1134–1149, Mar. 2021, doi: 10.1109/TCYB.2019.2962601.

[31] A. Chowdhury, G. Karmakar, J. Kamruzzaman, A. Jolfaei, and R. Das, "Attacks on self-driving cars and their countermeasures: A survey," *IEEE Access*, vol. 8, pp. 207308–207342, 2020, doi: 10.1109/ACCESS.2020.3037705.

[32] C. Zhao, J. S. Gill, P. Pisu, and G. Comert, "Detection of false data injection attack in connected and automated vehicles via cloud-based sandboxing," *IEEE Trans. Intell. Transp. Syst.*, early access, Jun. 23, 2021, doi: 10.1109/TITS.2021.3090361.