LIPs: A Protocol for Leadership Incentives for Heterogeneous and Dynamic Platoons

***Abstract*—With the ever-increasing problems of higher fuel costs and greater traffic congestion, shipping and long-distance travel via interstates and highways continues to become more expensive in terms of time and money. With the advent of semi- and fully-autonomous vehicles, platooning is designed to decrease the amount of fuel used and decrease the space between vehicles to help lower costs and reduce congestion. While much work has been done regarding predetermined platoons with homogeneous vehicles, less work has been done dealing with dynamic and heterogeneous platoons. Dynamic platooning with heterogeneous vehicles open a new horizon of problems with the introduction of several extra variables including dynamic platoon creation and management, untrusted users, differences in vehicle mechanics, and differences in fuel savings. One major problem facing dynamic, heterogeneous platooning is the leadership forfeiture abuse. Because there are currently no incentives for vehicles to lead in these platoons and the savings are much better when being a follower, it is more advantageous for a leader to forfeit their position and step down to become a follower and gain more benefits. In this paper, therefore, we propose a protocol, named as Leadership Incentives for Platoons (LIPs), which is suitable for dynamic platooning with heterogeneous vehicles. While the protocol provides a payment system that incentivizes individuals to lead, it is designed and implemented using the blockchain technology to offer a distributed secure environment for untrusted vehicles to interact. We demonstrate the application of the proposed protocol on a synthetic case study and evaluate the protocol by analyzing the time required for platooning operations/transactions as well as performing a usability test on a potential user group.**

***Index Terms*—Dynamic platooning, incentivization system, blockchain, untrusted users, autonomous vehicles.**

I. Introduction

Everyday, roads become more and more congested. Unfortunately, congested roads directly lead into other problems such as longer commute times, more fuel used, and a greater chance for vehicular accidents. If road congestion could be reduced even by a slight amount, it would save a large amount of money for anyone traveling. One of the solutions to the road congestion problem is platooning. In platooning, vehicles communicate either via a vehicular ad hoc network (VANET) or some other form of networking in order to drive much more closely together than would have been previously possible. Forming a "virtual train", follower vehicles receive logistics and management information from the leader. Whenever the leader accelerates, the rest of the platoon accelerates; likewise, whenever the leader brakes, the rest of the platoon brakes.

In the near future, vehicles will be able to join these platoons while on long trips in order to maximize their fuel savings. Due to the dynamic nature of this ability, many platoons will be working with a good amount of untrusted users at any one time. These users will also have their own motivations and goals that will not always align with maximizing the overall system goals. Given that it is more beneficial to be a follower in a platoon than the leader (e.g. in [1], [2]), it is not unlikely for vehicles to try and abuse this fact. In order to mitigate the effects of users acting selfishly and to make it desirable to remain as the leader of the platoon, an incentivization system is needed that is fair and can deal with untrusted users. Blockchain technology is very promising in this regard; it holds users to a higher standard of behavior through the use of a smart contract. This smart contract states what is appropriate behavior and blocks users from acting if it against that standard. Users do not have to trust each other, but they can trust the smart contract to hold them all accountable.

In this paper, we proposed the *Leadership Incentives for Platoons* (LIPs) protocol that was designed with heterogeneous and dynamic platoons in mind. The LIPs protocol deals with untrusted users by providing incentives for the leader of the platoon. Since it will be dealing with untrusted users, we chose blockchain as the backbone of the protocol to help with the financial transactions of this incentivized system as well as protect the platoon by ensuring that all platoon maneuvers are appropriate. To the best of our knowledge, the LIPs protocol is the first to provide an incentivization system for platoon leaders. The LIPs protocol is also capable of working on top of existing lower level protocols due to how it was designed. We also describe the leadership forfeiture abuse and on top of that, two case studies were performed that look into the feasibility, utility, and security of the LIPs protocol.

The rest of this paper is organized as follows: Section II describes the background of this work and platooning in general, Section III describes the LIPs protocol, Section IV provides the implementation details and two associated case studies, and Section V concludes the work with information about future work.

II. Platoon System Model

While platooning is becoming more and more popular in research, several distinct platooning configurations subcategorize the overall field. The field of platooning has several terms that either refer to very specific scenarios or even differ slightly between different works. Because of that, it is important to define the terms used in this work for clarity.

Vehicle Autonomy: The two major types of platooning autonomy are semi-autonomous and fully-autonomous. The semi-autonomous setup generally refers to a system in which vehicles not in a platoon are driven manually, but all vehicles

will become autonomous once they join the platoon if they are not the leader. In the fully-autonomous scenario, all vehicles are not driven manually as the name suggests. In [3], Nowakowski et al. show how this can be accomplished by utilizing *adaptive cruise control* (ACC) and *cooperative adaptive cruise control* (CACC). The LIPs protocol described in this paper can handle either of these types of platooning since it is on a higher-level than the protocol used for autonomy.

Platoon Formation: The next fork in the researching road regards how the platoons are formed. Static platooning is the simplest configuration, in which vehicles are predetermined to be members before the vehicles even leave for their journey. In this mode, a lot less variability manifests than in its counterpart–dynamic platooning. In dynamic platooning, convoys are formed ”on-the-fly”; vehicles are able to join and leave different platoons throughout their journey.

Platoon Membership Makeup: The third considerable dividing characteristic of platoon research regards the makeup of the vehicle types in the platoon. Homogeneous platoons are convoys in which all members are the exact same type of vehicles (i.e. same make and model). Heterogeneous platoons, like their name suggests, are made up of different types of vehicles, each with unique shapes, features, and capabilities. In this scenario, the order of the vehicles can have a great impact on how much fuel is saved due to differences in string aerodynamics. The spacing between the vehicles in the line is also of importance due to different braking abilities between vehicles.

Platoon Communication Strategies: The last differentiating factor deals with how information is dispersed throughout the platoon. In all scenarios that we have seen, each vehicle communicates back and forth with the *leading vehicle* (LV), which is also known as the platoon leader. One of the main information reception structures involves each *follower vehicle* (FV), or non-leading vehicle, receiving data from both the LV and the vehicle, which we will refer to as the *next vehicle* (NV), which is located directly in front of the FV. The other main communication configuration contains everything from the previous structure as well as the inclusion of information received from the vehicle, which we will refer to as the previous vehicle (PV), that is directly behind the FV. This second setup is more safe due to being aware of what the vehicle behind is doing. For example, if the PV starts accelerating suddenly, the FV is able to notice that and accelerate to match its speed and avoid the collision. An example of the two different communication structures can be seen in Figure 1.

While other communication formats are also out there (e.g. two predecessors following in [4]), these two appear to be the most promising because of a solid balance between safety and network overhead/throughput. Our protocol, again, exists on a higher level than these communication structures and is unaffected by which format is in place.

In terms of communication protocols, several different standards have been utilized in research. Ultimately, vehicle ad hoc networks (VANETs) will be how the vehicles communicate with each other, but ZigBee, WiFi, and 4G/5G can each be used as the underlying protocols of the VANET and for platooning specific functions.

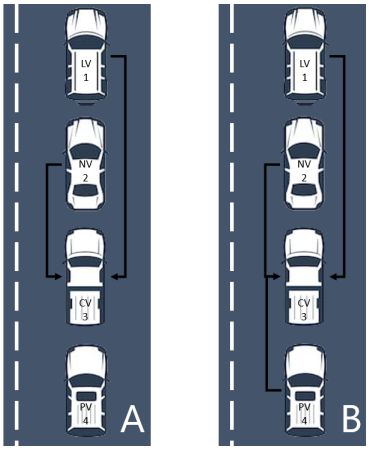

Fig. 1. Two common communication reception structures from the perspective of a following vehicle. In this case, we label the following vehicles we are focused on as the current vehicle (CV). In A, the CV listens to both the Lead Vehicle (LV) and the Next Vehicle (NV). In B, the CV listens to the Previous Vehicle (PV) in addition to the LV and the NV.

Platooning Interference: The term ”interfering vehicles” in the context of platooning refers to any vehicle that is not a member and that is traveling nearby in a way that prohibits or interferes with proper platoon manuevers (e.g. lane changes or vehicle adds). This does not necessarily have to be malicious in nature and, in practice, will most likely be a result of another vehicle overtaking the platoon or ignorantly merging in the middle of the platoon because a large enough space was present for whatever reason.

III. Related Works

In order to gauge the novelty of this paper's work, it is important to look into the other relevant research that has been accomplished.

Communication-based Protocols and Strategies: One very promising work, by Santini et al. in [5], provides a controller for platooning that allows for heterogeneous vehicles. The primary concern of their system was to ensure string stability and to maintain the appropriate inter-vehicle gap. Another communication-based solution, by Gheorghiu et al. in [6], deals with interfering vehicles by dynamically adapting the structure of the platoon to allow them to safely overtake it. Similarly, but for a different purpose, Liu et al. consider a flexible platooning problem in [7]. In their work, they use distributed model predictive control techniques to allow for dynamic joining and leaving of platoons.

For another type of communication, Jia et al. consider a model for the interactions between autonomous vehicles in a platoon and human-driven ones in [8]. They accomplish this through a consensus-based control algorithm that handles the differences between the two modes of vehicle control (i.e. autonomous and human-driven).

In [9], Baldi and Frasca mathematically synchronize the states of heterogeneous entities with unknown dynamics.

While they do not specifically make this for platooning, they do provide an example of how it can be used with platoons that utilize CACC. With a similar strategy in [4], Chehardoli and Ghasemi prove that the estimates of unknown values for aerodynamics (e.g. air drag, rolling resistance) converge to their actual values. They consider both communication strategies discussed in Figure 1 as well as an additional one in which vehicles also receive information from the predecessor of their predecessor.

Unlike the previous papers, Zheng et al. in [10] treat heterogeneity as a feature instead of an uncertainty in the problem formulation. In their research, they consider it from a cooperative platoon perspective in which interactions are acyclic and directed. Varada et al. in [11] also consider the control of cooperative heterogeneous platoons. The main focus of their paper was the design of a system that can improve the safety and stability of a platoon under delays. They also consider all of the different mechanical aspects of the vehicles as well as those communication delays in their simulations.

Platooning Fuel Efficiency: Switching gears from communication, several papers explore the various factors affecting fuel efficiency in platoons. Specifically in [12], Siemon et al. analyze the differences in aerodynamics with different truck and trailor configurations in platoons. This is accomplished through the use of computational fluid dynamics (CFD).

In another similar work, [13], Turri et al. propose methods to ensure fuel efficiency and track a reference speed for platoons that are cooperative. Turri et al. also provide a controller for non-cooperative platoons that checks the benefits from different types of management schemes. Another study, performed by Lammert et al. in [14], actually compiles multiple platooning tests (e.g. wind tunnel data) performed by other researchers and correlates the information discovered from each of those.

Blockchain and Platooning: Currently, few works look to combine the usefulness of blockchain specifically for platooning.In [15], Rowan et al. use blockchain as a method of message transportation. By utilizing hardware-based side-channels, they are able to securely manage a blockchain public key infrastructure between the vehicles in the platoon.

Concerned with improving the timeliness of blockchain, Calvo and Mathar in [16] mainly tackle the plan of decreasing the validation time for blockchain transactions. They do this through the use of micro-transactions and only publishing when a disagreement occurs. Other than that, they also consider the verification of vehicle identities.

In [17], Hexmoor et al. also consider applying blockchain to platooning in order to protect the members of the platoon and to share data rapidly. Furthermore, in [18], Lu et al. develop a reputation system for anonymous users in vehicular ad-hoc networks (VANET) with a blockchain-based approach.

Lastly, in a study of blockchain technology use cases [19], Lindberg illustrates how a blockchain system would work with platooning. Furthermore, Lindberg discusses the disparity among the savings of the platoon members and even mentions the possibility of a payment or rotation system in order to mediate that disparity. However, Lindberg does not follow through with the specifics nor the implementation; our work builds on top of this theoretical groundwork.

In summary, few works exist that effectively combine the benefits of blockchain with the possibilities of platooning. No one to our knowledge has considered and implemented our contributions before. In addition to that, our work can be implemented in conjunction with any of the aforementioned blockchain-based platoon systems as ours works at a higher level on top of theirs.

Savings Dispersion Models: Lastly, we will look at papers that try to disperse the overall savings of the platoon in a more equal manner. It does not appear that a lot of research has been done in this area under the context of platooning. In [20], Bhoopalam et al. provide a pretty extensive overview of the existing research, mainly showing similarities from other fields that could possibly be applied in platooning. They suggest sharing benefits by taking turns being the leader, even referencing an example of its application with UAVs in [21], but this is a solution only when one drives with the same vehicles on a regular basis. Furthermore, Bhoopalam et al. suggest other possibilities like using a reputation system to prevent mischievous behavior or formal contracts, but did not reference anyone doing anything similar in platooning.

Contributions: Unlike the existing literature, we propose an incentivization system for heterogeneous, dynamic platoons. Through the use of blockchain, we enact a smart contract that helps to prevent (and disincentivize) untrusted or unauthorized users from playing unfairly.

IV. Our Proposed Solution: LIPs Protocol

In LIPs protocol, we implement the platoon maneuvers through smart contracts. This allows a user to agree on the platoon management and payment system without trusting the other users. Any complex platooning maneuver can be broken down into four basic movements: the join, the leave, the merge, and the split. For example, the act of a vehicle leaving the third slot of a platoon of size four would break down into a split, another split, and a merge (e.g., Figure 2). In our protocol, we are only concerned with the upper-level logical and financial management; lower-level protocols will deal with the physical management of the platoon and pass that information upward to our protocol. In this section we first address the implemented maneuvers for platoon members, over Hyperledger Fabric. In the second part, we discuss all payment schemes in LIPs.

A. LIPs Maneuvers

So far we have implemented four maneuvers for each vehicle v_i who wants to interact with a platoon.

1) *Join Platoon*: First, we will discuss our implementation of the *Join Platoon* maneuver. To begin, the user will pass the ID of the platoon it wishes to join. If the user is already in a platoon, it will not be able to join another platoon without leaving its current one. Any vehicle should be able to join unless they owe too much money

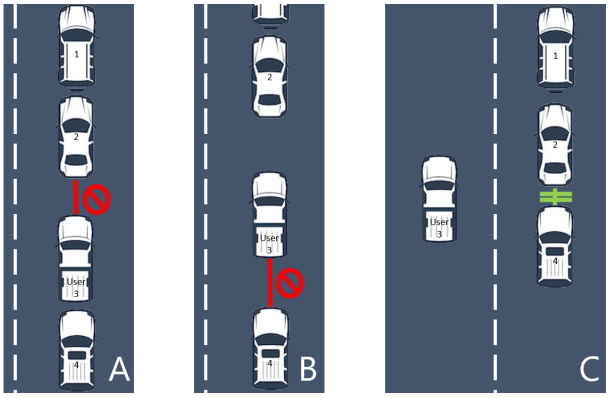

Fig. 2. A: User 3 splits with back half of platoon from front half. B: User 3 splits from back platoon. C: Back platoon merges with front platoon.

TABLE I
LIST OF SYMBOLS

Symbol	Definition
M_{v_i}	Vehicle v_i's base miles per gallon
g	Price for a gallon of fuel
x	Distance traveled (miles)
C_{v_i}	Fuel consumed since last platoon event by vehicle v_i
U_{v_i}	Fuel consumed if not in platoon by vehiche v_i
E_{v_i}	Fuel saved from platooning by vehicle v_i
A	Average percentage of fuel saved by platoon
S_{v_i}	Money saved from platooning by vehicle v_i
J_l	Fuel leader l would have consumed if he saved at average
N_l	Fuel leader l will be compensated for
R_l	Money leader l will be compensated for
V	Average monetary savings of followers
Y	Average follower payment
D_{f_i}	Money each follower f_i will play leader
Z_l	Leader l's total savings
Z_{f_i}	Total savings for follower f_i
B	Leadership bonus

and need to reconcile their debt, or their reputation score is too low. The only other time a user cannot join is if the platoon is already of maximum size, since platoons cannot be infinitely long and long platoons can cause serious traffic jams. It is at this point that the protocol waits for lower-level protocols to tell it everything is good to go. Once the user is accepted, the payments for the current followers of the platoon will be made to the platoon leader and the reputation scores will be adjusted appropriately. Also, the current variables tracking the platoon's history will be reset back. To elaborate on that through an example, values like the distance traveled in this formation will be set back to zero and will start counting up for new calculations since the platoon now has a new formation. At the end, the user will be added to the platoon.

2) *Leave Platoon*: Whenever a user wishes to exit, the *Leave Platoon* maneuver is invoked. This event should never be blocked, because one would be holding the vehicle hostage otherwise. Obviously, the code prevents a user from leaving a platoon it is not a member of. Once this process has started, each vehicle will pay the leader whatever it owes, reputation scores will be calculated, and the variables will again be reset. Depending on where the leaving vehicle is in the platoon, it may invoke a series of splits and/or merges in order to safely exit (e.g., Figure 2).
3) *Merge Platoon*: The *Merge Platoon* maneuver allows a platoon to combine with another platoon. Since merging is ultimately joining another platoon, the maneuver will be only be allowed if it does not set the resulting platoon above the vehicle limit. Before the merge, however, both platoons will need to process all of the payments to the leaders and calculate the resulting reputation scores for everyone. After this, the merge will be allowed if the reputation score of each vehicle in the joining platoon is high enough. This should be the case since the vehicles are already in a platoon; however, as a security practice, it is always best to double check. Also, both platoons might have had separate reputation requirements, so theoretically one could be in a platoon and not allowed in another. Once our protocol has been notified that everyone is in the proper position, the merging platoon will leave their current platoon and join the platoon being merged with.
4) *Split Platoon*: The last maneuver to be discussed is *Split Platoon*. This has elements of the Leave Platoon maneuver, but ultimately requires more work. The split is performed on a specific user who will become the leader of the new platoon, consisting of the vehicles behind user in the current platoon. First, every following vehicle pays the leader of the original platoon, and everyone's reputation scores are updated. Then, the second platoon is created and everyone from the new leader to the back of the original platoon are added to the new platoon and removed from the original platoon. It is at this time that the variables are reset for future calculations. The result is two separate platoons with their own leaders.

B. Payment Schemes

While designing this incentivization system, we considered many different *payment schemes*. For clarification, all of these schemes involve the following vehicles paying the leader for his services. This allows the leader to achieve near-equals benefits compared to the rest of the platoon. The overall goal is for all members of the platoon to save money through their participation. This goal must be achieved; failure to do so would make joining a platoon a waste of money. Table I summarizes the notations used throughout the paper.

The payment scheme used in the LIPs protocol is one in which each vehicle's payment is scaled based on how much money each individual is saving. Anytime a major change to the platoon occurs, i.e., a vehicle joins or leaves, our system takes in a set of inputs and calculates how much money each individual should pay the leader. The first input, M_{v_i}, is a vehicle v's standard miles per gallon for the same journey

without a platoon. We expect this variable to be reported by the vehicle's manufacturer when a certain make and model are given the appropriate platooning software and capabilities.

The next input is the price for a gallon of fuel g. We foresee this variable being provided via a query from the local fuel station prices. Also, it is important to note that this system will also work for electric vehicles as well; we just decided to choose gasoline as the fuel for the sake of familiarity. An interesting extension to our work would be calculating the payments with the inclusion of both gasoline and electric vehicles.

The third input necessary is the distance traveled x since the last platoon event. This value would be easy to determine given that all vehicles have an odometer. A harder value to determine is our last input C_{v_i}, which is the amount of fuel the vehicle consumed since the last platooning event. While these values could be reported directly from the vehicles themselves, that allows malicious users to attack the system in a variety of ways. For example, a user could lie about how much fuel was saved; a user could cause payment delays by not entering any value at all; and there is also the possibility of faulty monitoring equipment sending incorrect values to the smart contract. A better solution to this issue is to embed the fuel savings calculations into the smart contract itself using some set of standardized methods which would be previously researched and agreed upon by all parties involved in the smart contract creation. This allows all involved users the ability to audit the fuel savings calculations to verify that these calculations are indeed fair, and it also prevents malicious users from manipulating the system.

Now we can get an estimate of how much fuel each vehicle v_i would have used had it not been in a platoon by calculating U_{v_i} by,

$$U_{v_i} = \frac{x}{M_{v_i}}. \tag{1}$$

With this value, we are now able to determine the amount of fuel saved by each vehicle v_i, represented with E_{v_i}

$$E_{v_i} = U_{v_i} - C_{v_i} \tag{2}$$

Next, we calculate the percentage change in fuel consumption for each vehicle, denoted by ΔC_{v_i}

$$\Delta C_{v_i} = (1 - \frac{C_{v_i}}{U_{v_i}}) \times 100 \tag{3}$$

We designate the average percentage of fuel saved by each vehicle by ΔA

$$A = \frac{\sum_{v_i \in \mathcal{P}} \Delta C_v}{|\mathcal{P}|} \tag{4}$$

In which a platoon is represented by the set $\mathcal{P}$ that contains all vehicles v_i. We designate the amount of money each vehicle v_i saved through platooning by S_{v_i} and we can calculate it by,

$$S_{v_i} = E_{v_i} * g \tag{5}$$

With all of these preliminary calculations out of the way, we now start determining how much the platoon leader should be paid. First, we calculate how much fuel the leader l would have consumed had he received the platoon's average change in fuel consumption. This value is denoted by J_l,

$$J_l = U_l - \frac{A \times U_l}{100} \tag{6}$$

We then subtract that value from what was actually consumed by the leader l to find the amount of fuel that was not saved and that needs to be compensated for. This can be seen in the following equation denoted by N_l.

$$N_l = C_l - J_l. \tag{7}$$

If we take this result, we can then determine how much more money the leader l would have saved by not using that amount of fuel. We denote this value as R_l and can be calculated by

$$R_l = N_l \times g. \tag{8}$$

We will use this as the amount the leader l should receive from the followers in the platoon. Now that we have the value the leader l will receive from the platoon, we need to find out what each follower $f \in \mathcal{F}$ will owe where $\mathcal{F}$ is the set of all followers of the platoon $\mathcal{P}$. First, we find the average savings, V, of all followers $f \in \mathcal{F}$ by

$$V = \frac{\sum_{f \in F} S_f}{|F|} \tag{9}$$

Second, we find what the average payment Y of each follower $f \in \mathcal{F}$ would be using the amount of money the leader is owed R_l,

$$Y = \frac{R_l}{|\mathcal{F}|} \tag{10}$$

With this information, we find the amount of money each follower f_i will pay the leader, denoted as D_{f_i},

$$D_{f_i} = Y \times \frac{S_{f_i}}{V} \tag{11}$$

In summary, to find the leader l's total savings from this system, i.e., Z_l, we add his payment from the followers to his earlier savings from simply platooning.

$$Z_l = R_l + S_l \tag{12}$$

Similarly, we find the total savings for the followers Z_{f_i} through the following equation:

$$Z_{f_i} = S_{f_i} - D_{f_i} \tag{13}$$

This value should always be positive, otherwise the follower would have been better off not joining the platoon.

The previous equations have all been a part of our base payment model. We have included a couple extra options as extensions to the LIPs protocol for flexibility purposes. The first option we added is a bonus payment just for being the leader. In semi-autonomous platoons, being the leader still requires a human driver. It is also more risky to be the leader,

so providing a bit more incentives at times may be necessary. To account for this, a slight modification to the adjusted amount of fuel used by the leader was added. We add the leadership bonus B to the average percent of fuel saved by the platoon in the calculation to slightly shift the amount of money the leader will be paid as follows by changing Equation (6) to

$$J_l = U_l - \frac{(A+B) \times U_l}{100}. \tag{14}$$

The other option we provide with our system is to include a penalty for choosing to join and leave the platoon. Because of the cost of overhead incurred by changing the structure of the platoon while on the move, a payment would both decrease needless movements in and out of the platoon and compensate for the savings lost by the members of the platoon that are affected by the movement. With all of this in mind though, this payment would have to be fairly small; otherwise, no one would find it worth it to join the platoon in the first place.

In the following section, we navigate the security considerations that went into the LIPs protocol along with how we designed it to function.

V. Security Considerations

Protecting the drivers and their vehicles while platooning is of utmost importance. However, several security problems still linger among the various existing protocols. In the following subsections, we target three major cyber threats to platooning.

A. Untrusted Users

In the dynamic platooning scenario, various individuals and entities will be entering and leaving platoons as it travels. On long journeys and even shorter journeys, it is very feasible to meet someone for the first time, share a platoon with them, and then never see them again. In general, it is very hard to trust any individual that one has never worked with before when the risk is elevated. It is especially hard to trust someone immediately after being introduced.

By implementing blockchain with smart contracts, one does not necessarily have to trust the user. Smart contracts provide rules, penalties, and enforcement so that both parties can work together. It allows for the removal of a middle-man or trusted third-party, while still enabling different, untrusted users to participate in the system. In our case, all of the platoon actions and the payment system itself are both implemented through these smart contracts. Instead of trusting the other user, one can trust a verifiable system.

The LIPs protocol includes two systems to better help judge untrusted users. One of these reveals how often they are a leader compared with how often they are a follower. Those who only follow will have really low leadership reputation scores. The other system has to do with the users payment history. If they owe a certain amount of money, they will not be able to be a follower of a platoon until they pay their debts off.

Since the LIPs protocol deals with reputation scores, it is best to talk about the protocol's resistance to Sybil attacks.

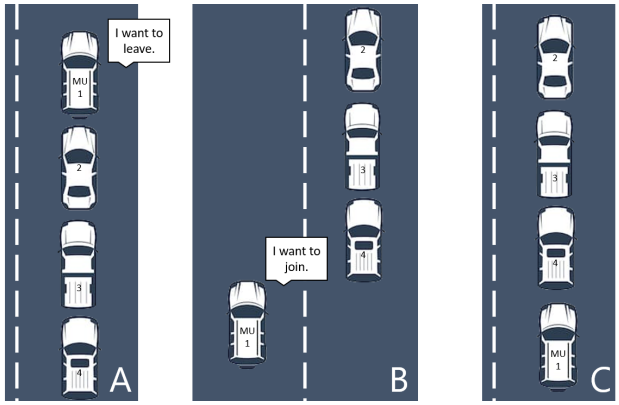

Fig. 3. A: Malicious user is discontent with receiving less savings than the followers, so it asks to leave. B: Malicious user moves to the back of the platoon and requests to join again. C: Malicious user is now saving more fuel at the cost of adding overhead to the platoon.

Because of how we implemented it, a user cannot just vote for himself up by creating fake accounts to vouch for him. Our system is based off driving history, so the only way to increase his score would be to form and lead platoons with his created users. We see this platooning system being tied to something like a license plate for verification, so trying to "boost your score" seems very unfeasible given how you would have to have accounts for a real, verified vehicle.

B. Repudiation

Another issue related with untrusted users is when they decide to refute true claims against them. They deny all allegations by saying they were not responsible for a certain problem or action that they did, in fact, have something to do with. This is commonly known as repudiation.

Again, because of implementing this protocol with blockchain, repudiation is no longer a problem. Since the blocks are immutable, any repudiation claim can be checked against the blockchain for verification. Any lie will be discovered. Since our smart contracts describe the payment system, the platoon actions, and reputation system, non-repudiation can be guaranteed for each of these.

C. Leadership Forfeiture Abuse

One of the underlying problems with platooning is that the followers in the platoon benefit much more than the leader (e.g. in [1], [2]). If this is the case, why would anyone want to lead? It would be much more profitable as an individual to simply leave the platoon he was in and rejoin it in the back so that he is now a follower. We call this process the "Leadership Forfeiture Attack" and provide an example of it in Figure 3. It is an attack in the sense that the individual is greedy and wants to maximize his profits, so he does what is best for him. The platoon suffers in the sense that there is overhead when the leadership changes hands. To bring this to the worst possible scenario, if each individual in the platoon performs this attack, they will endlessly be cycling and thus ruining the gains they could have been making. While this is not as big of an issue

when multiple, or all, vehicles in the platoon belong to the same organization and those platoons are created beforehand, it becomes especially problematic in the dynamic platooning scenario. If people can come and go as they please, it always makes sense to never lead and always follow if you want to maximize your benefits.

It is possible, however, to provide a major roadblock to this attack by providing incentives. If there is more beneficial behavior or even if leading provides roughly the same benefits as following, much less incentive remains to perform this action; it would hurt one to change positions in the platoon more in this case. In our protocol, we implement a payment system, protected through blockchain, that provides users with more incentive to not leave leadership. The details of how we do this can be seen in the following sections.

VI. Implementation and Case Study

To better illustrate the protocol in action, we have developed an implementation of LIPs as well as two case studies. The first case study looks at the protocol running under expected, ideal circumstances. The second study tries to exploit the protocol for malicious gain.

A. Implementation

LIPs has been fully implemented as described in this paper. The implementation is based on the Linux Foundation's Hyperledger Fabric framework, which is a blockchain-based smart contract platform [22]. Hyperledger Fabric was chosen for this project because it is a modular system which allows simple development of smart contract chaincode written in the GO language. Furthermore, Fabric is able to provide a plaform for private business-based blockchain solutions, which is ideal for the use case of LIPs.

1) Chaincode Summary

The core of LIPs is the GO chaincode for the Hyperledger fabric application. This code can be found on the project's Github page at the following address:

`https://github.com/Samboy218/PlatoonTest`

In LIPs, the chaincode defines all actions that members of the system can take, and it contains all of the application logic which enforces the rules agreed upon by the smart contract users. The following actions are defined by the smart contract:

- joinPlatoon, which allows a user to join a platoon or create a new platoon. This action will be denied if the user is already part of a platoon
- leavePlatoon, which allows a user to leave their current platoon. This action will be denied if the user is not currently part of a platoon
- splitPlatoon, which allows a platoon to split into two separate platoons. This action will be denied if the user is not currently part of a platoon
- mergePlatoon, which allows a platoon to merge into another platoon. This action will be denied if the user is not currently the leader of a platoon
- changeSpeed, which allows a platoon to record a change in speed. This action will be denied if the user is not currently the leader of a platoon

All of these functions are designed to allow the greatest flexibility in how users of the system wish to utilize the platooning, while at the same time ensuring that no malicious behavior can take place. Payment is calculated whenever there is a change in the members of a platoon, which ensures that payment is always collected from users.

The payment scheme is made to be as flexible as possible, and since the chaincode represents a smart contract, whatever method used for payment would be something that all users of the system have previously agreed to use. For testing purposes, the chaincode uses a predictive model to estimate how much fuel is saved based on different efficiency classes of vehicles. The payment scheme knows the expected MPG of a specific vehicle based on its efficiency class, and then the model uses a lookup table to find out what the vehicle's expected platoon MPG is based on what vehicle type is in front of it in the platoon. This model accounts for the fact that LIPs can handle heterogeneous platoons. In a production environment, this testing model may not be suitable for real conditions. Luckily, the chaincode that LIPs uses is modular, so it would be relatively simple for a different model to be used which is based on industry research or aerodynamic properties.

As mentioned earlier, the chaincode defines the logic of the application. This chaincode lives in a network which contains several different components that all work together to form the Hyperledger Fabric Network. In the Fabric network, there are many different nodes, and there are different types of nodes. Clients are nodes which submit transactions and transaction proposals. Peers keep and maintain a copy of the ledger, and they also commit transactions to the ledger. Peers can also be Endorsing Peers, which means that Peer has some role in determining what transactions are valid based on the execution of the chaincode. The last type of node is the Orderer. These nodes form the ordering service which is in charge of implementing delivery guarantees for transactions (atomic delivery, broadcast, etc.). The test environment for LIPs is composed of one organization, org1, which has a single orderer, two peers, and a single certificate authority. This network configuration was used throughout the testing of LIPs, but Hyperledger is able to scale this network configuration to include any number of organizations, peers, and orderers.

2) Data Flow

The flow of data in the Hyperledger application follows a simple process in order to ensure that all transactions are properly validated. First, a client who wishes to initiate a transaction will send a transaction proposal to the endorsing peers in the network. Endorsing peers contain the chaincode and the ledger state. The endorsing peers will run the chaincode against the proposed transaction, and then they return a signed transaction endorsement if the transaction was valid according to the chaincode. The client then sends the endorsed transaction to the ordering service. The job of the ordering service is to ensure that all peers get messages in the correct

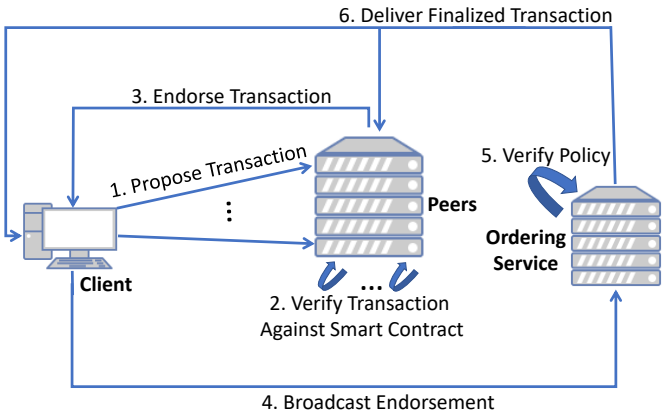

Fig. 4. Transaction flow model in blockchain (Hyperledger Fabric). The figure is adapted from [23].

order. When a peer receives an endorsed transaction from the ordering service it will update its ledger state based on the transaction. This process is shown in Figure 4.

In the LIPs implementation, each function that a user can take is coded as a separate chaincode function. This allows all actions to be clearly defined in the contract and if a change ever needs to be made the code allows for simple access to all of the chaincode functionality. The ledger state is designed to be easily manipulated by the chaincode while also remaining understandable to human observers. To this end, the data in the ledger is stored in the JSON format, with separate objects for users and platoons. The user data structure contains the ID, reputation, and data relating to the user's vehicle's fuel efficiency. User IDs are known to the chaincode and verified by parsing the X509 certificates provided by the certificate authority. Each platoon stores the ID, current speed, distance traveled, and the members of the platoon. All of this data (except user IDs) can be modified by executing the various chaincode functions. In some Blockchain applications, users have the ability to directly send messages to other users; in LIPs this is not necessary, so that functionality is not included in the application.

B. Case Study 1: Ideal Circumstances

The first case study looks at what happens under ideal operating circumstances in the platoon network. This test was done with 15 users, labeled u_1 through u_{15}, and each user represents a vehicle on the road. This case study was ran two times, once with a leader bonus of 2% and once with no leader bonus. The raw data output for this run can be found on the Github listed as $runLeaderBonus.json$ and $runNoBonus.json$ respectively. In Table II, we have listed the transactions that were used for the scenario along with their resulting platoon membership statuses.

Listed in Table III are the net changes in wallet for the transactions in Case Study 1. A positive value means that the user received money, while a negative value shows that the user paid money.

TABLE II
CASE STUDY 1: ORDER OF EVENTS

Action	Leader: Paid By	Platoon Members After Action
u_1 joins P_1		P_1: u_1
u_2 joins P_1		P_1: u_1, u_2
u_3 joins P_1	u_1: u_2	P_1: u_1, u_2, u_3
u_4 joins P_1	u_1: u_2, u_3	P_1: u_1, u_2, u_3, u_4
u_5 joins P_1	u_1: u_2, u_3, u_4	P_1: u_1, u_2, u_3, u_4, u_5
u_6 joins P_1	u_1: u_2, u_3, u_4, u_5	P_1: u_1, u_2, u_3, u_4, u_5, u_6
u_{11} joins P_2		P_1: u_1, u_2, u_3, u_4, u_5, u_6 P_2: u_{11}
u_{12} joins P_2		P_1: u_1, u_2, u_3, u_4, u_5, u_6 P_2: u_{11}, u_{12}
u_{13} joins P_2	u_{11}: u_{12}	P_1: u_1, u_2, u_3, u_4, u_5, u_6 P_2: u_{11}, u_{12}, u_{13}
u_4 splits P_1 P_3 created	u_1: u_2, u_3 , u_4, u_5, u_6	P_1: u_1, u_2, u_3 P_2: u_{11}, u_{12}, u_{13} P_3: u_4, u_5, u_6
u_5 splits P_3 P_4 created P_3 ended	u_4: u_5, u_6	P_1: u_1, u_2, u_3 P_2: u_{11}, u_{12}, u_{13} P_4: u_5, u_6
P_4 merge: P_1 P_4 ended	u_5: u_6 u_1: u_2, u_3	P_1: u_1, u_2, u_3, u_5, u_6 P_2: u_{11}, u_{12}, u_{13}
u_6 leaves P_1	u_1: u_2, u_3, u_5, u_6	P_1: u_1, u_2, u_3, u_5 P_2: u_{11}, u_{12}, u_{13}
P_2 merge: P_1 P_2 ended	u_{11}: u_{12}, u_{13} u_1: u_2, u_3, u_5	P_1: u_1, u_2, u_3, u_5, u_{11}, u_{12}, u_{13}
u_4 joins P_1	u_1: u_2, u_3, u_5, u_{11}, u_{12}, u_{13}	P_1: u_1, u_2, u_3, u_5, u_{11}, u_{12}, u_{13}, u_4
u_{12} splits P_1 P_5 created	u_1: u_2, u_3, u_5, u_{11}, u_{12}, u_{13}, u_4	P_1: u_1, u_2, u_3, u_5, u_{11} P_4: u_{12}, u_{13}, u_4
u_2 splits P_1 P_6 created	u_1: u_2, u_3, u_5, u_{11}	P_1: u_1 P_4: u_{12}, u_{13}, u_4 P_6: u_2, u_3, u_5, u_{11}
u_3 splits P_6 P_7 created P_6 ended	u_2: u_3, u_5, u_{11}	P_1: u_1 P_4: u_{12}, u_{13}, u_4 P_7: u_3, u_5, u_{11}
P_7 merge: P_1 P_7 ended	u_3: u_5, u_{11}	P_1: u_1, u_3, u_5, u_{11} P_4: u_{12}, u_{13}, u_4
u_2 joins P_4	u_{12}: u_{13}, u_4	P_1: u_1, u_3, u_5, u_{11} P_4: u_{12}, u_{13}, u_4, u_2

TABLE III
NET WALLET CHANGES FOR EACH USER

User	Case Study 1	
	No Bonus	Bonus
u_1	.7173	1.1955
u_2	-.0844	-.1548
u_3	-.3052	-.5218
u_4	-.0238	-.0316
u_5	-.1107	-.1741
u_6	-.0864	-.1459
u_{11}	.0696	.1245
u_{12}	-.0629	-.1149
u_{13}	-.1134	-.1768

C. Case Study 2: Malicious Behavior

For the second case study, we looked at how the protocol would react to invalid transactions and misbehavior. There are many ways a malicious user could attempt to use the network in a way which benefits them at the expense of the other users in the system. LIPs is designed to minimize the amount of ways an attacker could do this. For example, treat u_7 as a malicious user. u_7 wants to save more money than everyone else, but thankfully he will save roughly as much as everyone else. For example, u_7 could attempt to lie about how much gas he saved, but, because LIPs uses a predictive model which is stored in the chaincode, u_7 actually has no way to manipulate those calculations.

Furthermore, u_7 believes that the leader of a platoon receives less fuel savings than the other members of the platoon, so it decides that it will never choose to lead a platoon. And if it ever is stuck as a leader, it simply leave the platoon and

then rejoins it in order to be put in the back for better savings; we refer to this methodology as the "Leadership Forfeiture Abuse". This plan will also be foiled through the LIPs protocol because of the incentive-based payment scheme. Even if u_7 never leads a platoon and thus saves more money on fuel than the leader, it will still be required to pay the leader an amount proportionate to its savings.

Since this plan did not work either, u_7 then may attempt to attack the network by joining platoons under another user's name to make the other user pay his platooning fees. This attack will fail because users in Hyperledger Fabric are authenticated using a certificate authority, and u_7 only has his own certificate. Finally, u_7 decides to simply fill the ledger with bogus transactions. u_7 attempts to leave a platoon when he isn't currently in a platoon, or merge with a platoon when he isn't the leader of a platoon. However, all of these transactions will be rejected by the endorsing peers because those transactions do not satisfy the smart contract. u_7 is unsuccessful in causing mayhem in the network.

A test scenario was created that is the same as the first case study, but included are six additional invalid transactions made by a malicious user. The invalid transactions are blocked, and the only thing that is different is that it there is additional time spent wasted checking the validity of the transactions.

VII. Evaluation

To evaluate the protocol further, the following subsections describe the performance testing and usability of the LIPs protocol.

A. Performance Testing

Another important aspect of a successful application is its ability to handle any demand in resources that could be expected under normal and abnormal usage conditions. To test the Hyperledger Fabric framework in this manner, the application was modified to allow the queuing of many transactions at once. Once the application processess all of these transactions the logs produced can be analyzed in order to find out where any performance bottlenecks may exist.

For this test, 400 transactions were submitted to the blockchain system. Each of these transactions causes the Fabric network to follow a lengthy process in proposing, verifying, endorsing, executing, and finalizing the transaction in order to commit some state change to the ledger. Each stage of the transaction is logged by the network along with timestamps, so the time each stage takes per transaction can be calculated.

This test showed that the average time per transaction was 5.41 seconds. Of this time over 90% was spent in the endorsement stage. From examining the logs of the test it was uncertain what exactly caused this bottleneck, but this issue would likely render Hyperledger Fabric not sufficient for large networks if it is not fixed in future versions; however, it is also a possibility that the extreme time spent in the endorsement stage is a result of a configuration in the fabric network that can be mitigated, and if that is the case then more research can be done into creating efficient networks which mitigate this bottleneck.

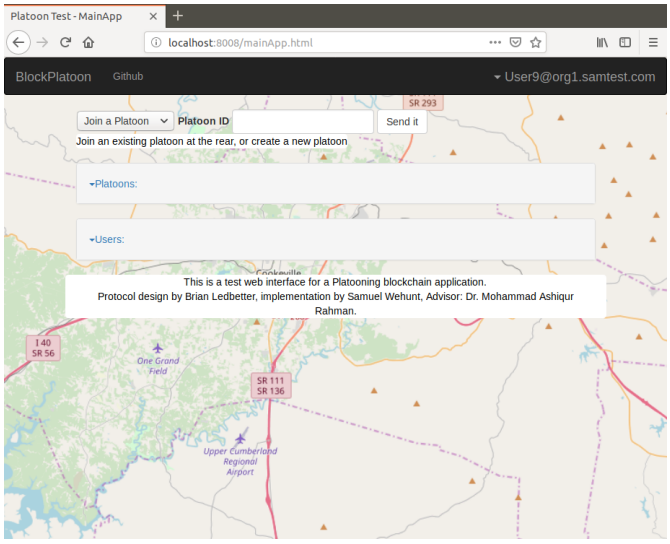

Fig. 5. Screenshot of LIPs Application

B. Usability

When making an application that end users will interact with, it is important to asses how usable that application is to a wide variety of audiences. To evaluate the usefuleness of the LIPs framework, a survey was given with 10 questions related to the application (as seen in Figure 5):

- Q1: Are the commands intuitive?
- Q2: Is it easy to navigate?
- Q3: Is it aesthetically appealing?
- Q4: Does it respond quickly to commands?
- Q5: Does it do what is expected?
- Q6: Does it provide enough information?
- Q7: If an error occurs, is it explained appropriately?
- Q8: How familiar are you with platooning?
- Q9: Could you see something similar to this working in the real world?
- Q10: Overall, how do you rate this application?

These questions were chosen with the goal of gaining a broader view of how users feel about the application and what needs improvement.

11 subjects were asked to use the application for a period of around 5 minutes, and then take a survey based on their experience. The results of this study are presented in Figure 6.

All in all, the application received fairly positive reviews. The largest variance came from questions seven and eight. While question eight is not in our control because it has to do with the user's background, we can improve our error messaging fairly easily. Currently, it is a little too verbose for the average user.

For this survey, we provided the subjects with a testing version of the application. With it, they were able to switch between users in the platoon and to try different platooning

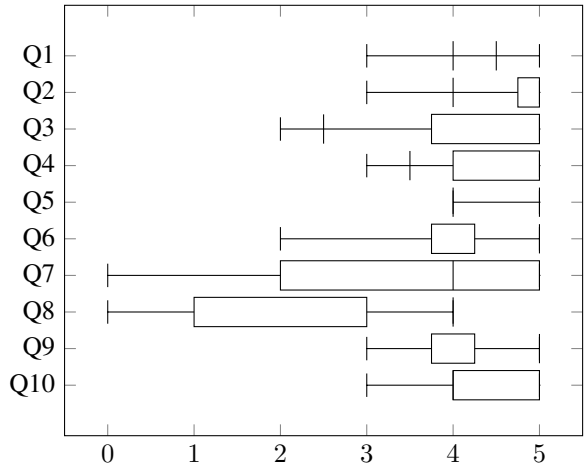

Fig. 6. Graph showing the distribution of points, on a scale from 0 to 5, for each question.

maneuvers. Going forward, this will need to be changed to only allow the user to control its own vehicle. We would also like to test a scenario with multiple users making requests at the same time in the same network.

VIII. Conclusion and Future Work

With the growth of autonomous vehicle technology, platooning is becoming an emerging concept to decrease the fuel usage as well as reduce congestion. While much work has been done regarding predetermined platoons with homogeneous vehicles, dynamic platooning with and heterogeneous vehicles is the ultimate future. However, the trusted management of such dynamic platoons while dealing with issues like the overhead of playing the leadership role must be addressed. In order to provide incentives to leaders of platoons in a heterogeneous and dynamic configuration, we proposed the LIPs protocol. Backed by blockchain, the LIPs protocol ensures that untrusted users will work together according to the smart contract, preventing mischievous behavior and unnecessary overhead. Furthermore, the leadership forfeiture abuse has been identified and counteracted against through the LIPs framework. This framework was tested in two scenarios and worked as expected, even in the midst of malicious behavior.

This research has several future directions. Firstly, we would like to calculate how much fuel each vehicle used more accurately than by just assuming the savings based on predetermined tables for the calculation. Finding these values in a similar manner to [12], seems to be the first logical step. Another route would be to use the logic in [4] to see if the estimated fuel used converges to its actual value. Secondly, we would like to use the information gathered to optimize the ordering of the platoon through smart contracts in order to increase savings for all members. Thirdly, we would like to be able to simulate the LIPs framework under heaver stress testing. Lastly, we would like to explore utilizing the LIPs protocol on top of existing protocols to gauge its operability with those lower-level protocols.

References

[1] A. Alam, B. Besselink, V. Turri, J. Martensson, and K. H. Johansson, "Heavy-duty vehicle platooning for sustainable freight transportation: A cooperative method to enhance safety and efficiency," *IEEE Control Systems*, vol. 35, no. 6, pp. 34–56, 2015.

[2] M. P. Lammert, A. Duran, J. Diez, K. Burton, and A. Nicholson, "Effect of platooning on fuel consumption of class 8 vehicles over a range of speeds, following distances, and mass," *SAE International Journal of Commercial Vehicles*, vol. 7, no. 2014-01-2438, pp. 626–639, 2014.

[3] C. Nowakowski, S. E. Shladover, X.-Y. Lu, D. Thompson, and A. Kailas, "Cooperative adaptive cruise control (cacc) for truck platooning: Operational concept alternatives," 2015.

[4] H. Chehardoli and A. Ghasemi, "Adaptive centralized/decentralized control and identification of 1-d heterogeneous vehicular platoons based on constant time headway policy," *IEEE Transactions on Intelligent Transportation Systems*, 2018.

[5] S. Santini, A. Salvi, A. S. Valente, A. Pescapé, M. Segata, and R. L. Cigno, "A consensus-based approach for platooning with intervehicular communications and its validation in realistic scenarios," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 3, pp. 1985–1999, 2017.

[6] R. A. Gheorghiu, V. Iordache, and A. C. Cormos, "Cooperative communication network for adaptive truck platooning," 2017.

[7] P. Liu, A. Kurt, and U. Ozguner, "Distributed model predictive control for cooperative and flexible vehicle platooning," *IEEE Transactions on Control Systems Technology*, 2018.

[8] D. Jia, D. Ngoduy, and H. Vu, "A multiclass microscopic model for heterogeneous platoon with vehicle-to-vehicle communication," *Transportmetrica B: Transport Dynamics*, pp. 1–25, 2018.

[9] S. Baldi and P. Frasca, "Adaptive synchronization of unknown heterogeneous agents: An adaptive virtual model reference approach," *Journal of the Franklin Institute*, 2018.

[10] Y. Zheng, Y. Bian, S. Li, and S. E. Li, "Cooperative control of heterogeneous connected vehicles with directed acyclic interactions," *arXiv preprint arXiv:1805.04304*, 2018.

[11] K. Varada, "Distributed cooperative control of heterogeneous multi-vehicle platoons," 2017.

[12] M. Siemon, "A numerical analysis of heterogeneous and homogeneous truck platoon aerodynamic drag reduction," 2018.

[13] V. Turri, "Look-ahead control for fuel-efficient and safe heavy-duty vehicle platooning," Ph.D. dissertation, KTH Royal Institute of Technology, 2018.

[14] M. Lammert, K. Kelly, and J. Yanowitz, "Correlations of platooning track test and wind tunnel data," NREL Technical Report NREL/TP-5400-68897, National Renewable Energy Laboratory, Tech. Rep., 2018.

[15] S. Rowan, M. Clear, M. Gerla, M. Huggard, and C. M. Goldrick, "Securing vehicle to vehicle communications using blockchain through visible light and acoustic side-channels," *arXiv preprint arXiv:1704.02553*, 2017.

[16] J. A. L. Calvo and R. Mathar, "Secure blockchain-based communication scheme for connected vehicles," in *2018 European Conference on Networks and Communications (EuCNC)*. IEEE, 2018, pp. 347–351.

[17] H. Hexmoor, S. Alsamaraee, and M. Almaghshi, "Blockchain for improved platoon security," *International Journal of Information*, vol. 7, no. 2, 2018.

[18] Z. Lu, Q. Wang, G. Qu, and Z. Liu, "Bars: a blockchain-based anonymous reputation system for trust management in vanets," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. IEEE, 2018, pp. 98–103.

[19] J. Lindberg, "Blockchain technology in scania services: An investigative study of how blockchain technology can be utilized by scania," 2017.

[20] A. Kishore Bhoopalam, N. Agatz, and R. A. Zuidwijk, "Planning of truck platoons: a literature review and directions for future research," 2017.

[21] D. Richert and J. Cortés, "Optimal leader allocation in uav formation pairs under no-cost switching," in *American Control Conference (ACC), 2012*. IEEE, 2012, pp. 3297–3302.

[22] "Hyperledger fabric," https://www.hyperledger.org/projects/fabric, accessed: 2019-01-03.

[23] O. Choudhury, H. Sarker, N. Rudolph, M. Foreman, N. Fay, M. Dhuliawala, I. Sylla, N. Fairoza, and A. K. Das, "Enforcing human subject regulations using blockchain and smart contracts," *Blockchain in Healthcare Today*, 2018.