VANET Clock Synchronization for Resilient DSRC Safety Applications

Sherif Hussein and Axel Krings
Computer Science Department
University of Idaho
Moscow, Idaho 83843-1010
huss3426@vandals.uidaho.edu, Krings@uidaho.edu

Azad Azadmanesh
Computer Science Department
University of Nebraska at Omaha
Omaha, NE 68182-0500
azad@unomaha.edu

***Abstract*—Vehicular Ad Hoc Networks (VANET) are the fast-changing networks for connected vehicles, in which Vehicle-to-Vehicle and Vehicle-to-Infrastructure communication are the basis for technologies aiming at reducing accidents and improving operation. DSRC Safety Applications, designed to assist drivers in order to avoid accidents, might be subjected to malicious attacks, such as GPS time spoofing attacks, which attempt to prevent time synchronization between vehicles. Failure of the centralized GPS-based clock synchronization has the potential to cause safety applications to fail. Therefore, decentralized clock synchronization can be a valuable approach for augmenting GPS-based clock synchronization.**

In this paper, a decentralized clock synchronization protocol for VANET is presented. The proposed protocol, based on approximate agreement, does not require any extra hardware nor modifications of any standards. The protocol was simulated using NS-3, and the results were analyzed and compared with previous synchronization protocols. The benefits of the proposed clock synchronization algorithm are higher resilience of safety applications to GPS spoofing attacks and when GPS signals are not available, such as in urban cities.

I. Introduction

Vehicular Ad Hoc Network (VANET) is a special type of Mobile Ad Hoc Network (MANET), where communicating nodes are moving vehicles. It assumes a dynamic topology due to the high mobility of participating vehicles, which may result in connections of short duration. Dedicated Short Range Communications (DSRC) offers the wireless support for Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communication to form a VANET [1]. It requires that each vehicle in the VANET is equipped with an On Board Unit (OBU) and each intersection has a Road Side Unit (RSU). One of the primary goals of VANET is to enhance road safety by employing several types of DSRC Safety Applications. Forward Collision Warning (FCW) and Emergency Electronic Brake Lights (EEBL) are examples of such safety applications that alert a driver about possible crash scenarios ahead. VANET allows vehicles to exchange their information like position, time, heading and break status with neighbors. A Global Positioning System (GPS) receiver attached to the OBU in each vehicle is the main source for such information. Based on the information exchange, DSRC Safety Applications alert drivers about road hazards with the goal of reducing the number of accidents. To assure data consistency and reliability, safety applications require that all vehicles be approximately synchronized to the same clock value.

Clock synchronization in wireless communication can be achieved using either centralized or decentralized approaches [2]. The current configuration of VANET uses GPS as a central controller for clock synchronization. However, GPS might be subject to frequent GPS signal outages in urban cities or subject to GPS spoofing [3]. Imagine an attacker launches a GPS spoofing attack near areas that suffer from frequent GPS signal outages, e.g., tunnels or parking garages in big cities, in coordination with the introduction of a road hazard. Such an attack can lead to serious clock synchronization problems between participating vehicles in VANET. For example, clock synchronization problems may force the EEBL safety application to discard messages that contain real warnings about road hazard because they could be considered too old (outdated). In this case, EEBL would fail to alert drivers not having direct visual contact to the hazard, thus potentially leading to rear-end collisions. To mitigate the centralized GPS clock synchronization as a single point of failure, one approach is to require all vehicles to participate in time synchronization. This requires vehicles to exchange and cooperatively agree upon their respective local clock values.

Agreement can be generally categorized into exact and inexact agreement. In exact agreement, all communicating vehicles are required to reach the same exact decision. Byzantine Consensus [4] and the Interactive Consistency Problem [5] are the best-known forms of exact agreement. On the other hand, in inexact agreement, also called Approximate Agreement (AA), vehicles are not required to reach agreement (to vote) on the same exact value. Rather, they must converge on final values that are within a predefined tolerance. AA must satisfy Agreement and Validity conditions: 1) the agreement condition requires that all non-faulty clocks halt with voted values that are within a predefined tolerance of each other; 2) the validity condition ensures that the final voted values stay within the range of the initially correct clock values. Most of the AA algorithms published employ rounds of data exchange and require the use of an approximation function F to update the clock values, which are then used in the next round of data-exchange. The objective is to gradually shrink the *diameter* (difference) between local values, by providing a sufficient number of rounds of data-exchanges to reach the predefined tolerance value.

The rest of the paper is organized as follows. Section II presents some necessary background and the problem definition. A brief summary of recent clock synchronization protocols are found in Section III. Section IV describes the network and fault model. The proposed clock synchronization protocol is presented in Section V. Simulation results and analysis are provided in Section VI, followed by conclusions in Section VII.

II. Background and Problem Definition

Communication in VANET uses DSRC [8]. The Federal Communications Commission (FCC) approved 75 MHz of bandwidth at 5.9 GHz for DSRC communication [9], which implements seven channels categorized into one Control Channel (denoted by CH178), and six Service Channels (CH172, 174, 176, 180, 182, and 184). The most important channel for this research is CH172, which is dedicated to safety applications.

A. DSRC Safety Applications

Various DSRC Safety Applications have been described in [10]. These applications rely on beacon messages called Basic Safety Message (BSM) [1], which are periodically broadcast by each vehicle every 100ms to exchange their vehicle information with their neighbors. The beacons contain vehicle-specific information like a vehicle identification (VID), location, motion information, and brake status, as well as a timestamp, which is the local clock value of the sender. The *timestamp* is the most important field in the BSM with respect to the proposed clock synchronization protocol, which will be presented using the EEBL safety application. The EEBL alerts drivers about hard braking events by vehicles in front of them moving in the same direction. This is very valuable in conditions with limited visibility, e.g., fog, or when the line of sight between two vehicles is obstructed.

Figure 1 shows the timing model of the EEBL. Assume that there are two vehicles separated by a short distance moving in the same direction in a one lane highway. The front vehicle, i.e., the Remote Vehicle (RV), is followed by the Host Vehicle (HV) whose OBU is running the EEBL safety application. Assume that the RV brakes hard at time t_{brake}, due to a road hazard. This “hard breaking” event is broadcast in subsequent BSMs. Upon receiving a BSM indicating the event, the EEBL application of the HV alerts its driver to the situation. As long as the alert is given before time t_{react}, the driver will be able to react and avoid a potential collision.

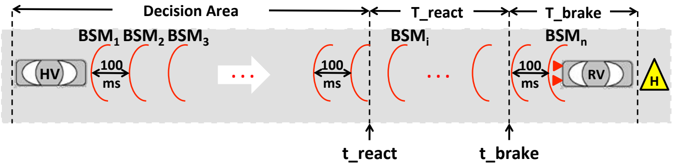

Fig. 1. EEBL timing model [11]

B. Problem Definition

According to [12], the BSM time-to-live, which is the difference between the *timestamps* of the HV and RV, should be no more than 500ms, and any BSM time-to-live exceeding this time should be considered outdated. Thus, VANET clock synchronization is critical, especially in the presence of malicious attacks or clock drifts. Since the GPS service is centralized, the OBUs should be able to use GPS clock values as a *reliable time source*, thereby allowing OBUs to periodically resynchronize their clocks. However, GPS may be subjected to frequent GPS signal outages in urban cities and/or are vulnerable to GPS spoofing attack like those described in [3].

In our own experience, GPS signal outages were noticed while conducting lab experiments using Arada Locomate OBUs [13]. For example, in one instance, while conducting experiments during a GPS reception outage, a difference of 2 seconds was observed between the clocks of two different OBUs.

Given the criticality of timeliness of BSM messages [12], the objective of this research is to design an alternative decentralized clock synchronization in case of GPS signal outages or GPS time spoofing attacks, thereby enhancing the resilience of DSRC safety applications.

III. Related Work

Clock synchronization protocols are classified into centralized and decentralized approaches [2]. The Global Navigation Satellite System (GNSS) [14] is a centralized clock synchronization protocol that is often used by ad-hoc networks. It has the advantage of simplicity, but requires the use of GPS data in the clock synchronization process. The decentralized approaches to clock synchronization require that all nodes (vehicles) participate in the synchronization process. Much research has been conducted on clock synchronization in ad-hoc networks, with less focus on VANET, as described below.

A. Clock Synchronization in Ad-hoc Networks

A Reference Broadcast Synchronization (RBS) protocol was proposed in [16] that exploits the broadcast property of the wireless communication medium. In this protocol, a transmitter sends a reference packet to all its neighbors. Each receiver records the receiving time of the reference packet according to its local clock and then exchanges the receiving time with other receivers. Based on receivers’ observation, the clock offset between the receivers can be easily computed. In [17], the Average Time Synchronization (ATS) protocol was proposed, which uses cluster heads as the key nodes in the network synchronization process. Each cluster head broadcasts a synchronization time packet to all nodes in the cluster. Each node in the cluster replies with the receiving time of the synchronization packet. The cluster head then averages all receiving times to get the current global time. Finally, the cluster head broadcasts a time message that contains the computed global time to every node. Both, the ATS and RBS protocols require a large number of message exchanges, and the synchronization process needs to be restarted whenever a new node with a different time joins the group. Network-wide synchronization is achieved by a similar approach among the cluster heads.

A dynamic wireless mobile network clock synchronization protocol was proposed in [15]. The protocol is based on linear approximate consensus in the presence of Byzantine faults. Periodically, each non-faulty node collects the timestamps sent by its neighbors and updates its clock value with the average of these timestamps. The authors assume the facilitating nodes are not capable of using fake identifiers. As demonstrated later, the average convergence function may not be the most efficient function in all conditions. Due to the high dynamic nature and fast network topology changes of VANETs, none of the previous protocols are suitable for vehicular ad hoc networks.

B. *Clock synchronization in VANET*

A clock synchronization protocol for VANET called Converging Time Synchronization (CTS) was proposed in [2]. This protocol is based on a sponsor election mechanism. One of the network nodes (an initiator) asks its neighbors to send their number of synchronized group members, their VIDs and time differences. Based on each neighbor's reply, the node that is synchronized to the largest number of neighbors is elected to be the new sponsor. The new sponsor then broadcasts a clock adjusting message to all nodes to adjust their own time.

In [19], a Hybrid Clock Synchronization (HCS) was proposed that included wireless sensors at fixed distances in the road to improve VANET synchronization. The main disadvantage in HCS is that it has large hardware overhead. The general equations that calculate the elapsed clock synchronization time for HCS and CTS can be found in [19] and [2] respectively.

A Time Table Diffusion (TTD) synchronization protocol was proposed in [20] that is immune to network topology and does not require the use of GPS. The protocol exploits the idea of the Time Table Transfer (TTT) protocol [21], where each node collects time information from its neighbors and constructs a time table. This time table is then broadcast to the neighbors to build their own time table and adjust their clock.

The aforementioned VANET clock synchronization approaches require a large number of extra messages in their synchronization protocols and do not follow the standard protocol of using BSMs.

IV. Network and Fault Models

Due to the properties of VANET, such as the fast changing, unknown topology and the number of participating nodes, the problem of distributed clock synchronization is very complex. We therefore take the step of introducing a simple model, representing the key properties of pathological, worst case, scenarios.

A. *Network Model and Notation*

This subsection introduces a simple VANET model to demonstrate the impact of connectivity on clock synchronization. For the sake of clarity, the focus is only on investigating the clock synchronization problem. Therefore, no message losses or collisions are assumed to occur, e.g., due to the hidden terminal problem or natural phenomena like shadowing. The terms “vehicle” and ”node” will be used interchangeably while describing the network model.

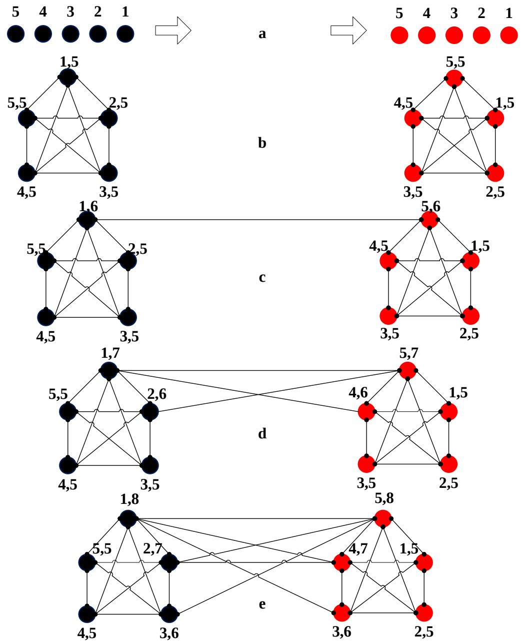

Fig. 2. Two cluster connectivity graphs

Assume N vehicles equipped with non-faulty OBUs travel on a road. Vehicles are moving in the same heading (direction) with fixed speed. The communication capabilities between vehicles are described by a connectivity graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of vehicles $\mathcal{V} = \{v_1, v_2, ..., v_N\}$, and $\mathcal{E}$ is the set of undirected edges $e_{i,j}$ for each communicating pair v_i, v_j. Let R be a given transmission range. Then, for each $e_{i,j} \in \mathcal{E}$, the distance between v_i and v_j, denoted by $d_{i,j}$, is less than or equal to R. Therefore, $\mathcal{E} = \{e_{i,j} | i \neq j, d_{i,j} \leq R, \forall i, j = 1, 2, ..., N\}$. Note that the graph will change over time as vehicles move. However, the graph is only used for the problem description and analysis. The graph does not have to be maintained as part of the algorithms below.

Assume that vehicles are divided into two fully connected clusters $\mathcal{V}_{C1}$ and $\mathcal{V}_{C2}$, where $|\mathcal{V}_{C1}| = |\mathcal{V}_{C2}| = N/2$. For simplicity, the distance between any two successive vehicles in the same cluster is fixed, e.g., to represent the road safety distance. Figure 2a shows the order of moving vehicles in each cluster. The distance between clusters, $d_{CL(1,2)}$, is defined as the distance between the leading vehicle v_1 in the left cluster (black) and the last vehicle v_5 in the right cluster (red) in the figure. As the cluster distance $d_{CL(1,2)}$ decreases, communication links between the isolated clusters in Figure 2b appear. For example, the connectivity graph shown in Figure 2c depicts the scenario in which vehicle v_1 of the left cluster and vehicle v_5 of the right cluster receive each other's messages. In the graph, vertices are labeled by a pair x, y, where x indicates the position of vehicle in its cluster and y is the size of the neighborhood of v_x, i.e., the number of neighbors including itself. For instance, label $1, 6$ represents v_1 as the first vehicle in the cluster with a neighborhood of size 6, composed of five neighbors and itself. As the clusters get closer, the neighborhoods of the closing vehicles increase, as can be seen in Figures 2d and e. It will be shown later that the size of the neighborhoods has implications on the clock synchronization speed.

Each vehicle v_i broadcasts a BSM every 100ms with a fixed transmission power P. No vehicle is assumed to be misbehaving, i.e., all vehicles follow the 10 BSMs/sec transmission rate, and no vehicle tampers with BSMs. Each vehicle maintains a sorted multiset in which it collects all timestamps, which will be used during a round-based voting process. Specifically, each vehicle v_i collects the *timestamp*, $t_{RV(j)}$, extracted from every BSM received from v_j, together with the corresponding local *reception time*, $t_{rec(j)}$, and saves them as pairs $(t_{RV(j)}, t_{rec(j)})$ in a table. Just before voting at time t_{vote}, these timestamps are adjusted, similar to [15], and included in a voting multiset $\mathbf{V}_i$. Each vehicle v_i has its own local voting multiset $\mathbf{V}_i = \{t_{k(1)}, t_{k(2)}, ..., t_{k(n_i)}\}$, where n_i is the neighborhood size of vehicle v_i and $k(j), j \leq n_i$ is the vehicle from which the time is logged. The time estimation value $t_{k(j)}$ is the clock value of v_j corrected to the time at v_i. Thus, $t_{k(j)} = t_{RV(j)} + (t_{vote} - t_{rec(j)})$.

Let $\mathbf{U}_{all}$ be the multiset of all clock values held by the vehicles in the network. Define $\delta(\mathbf{U}_{all})$ as the global diameter between clock values in the entire network. The global diameter is the difference between the maximum and minimum clock values in the network, i.e., $\delta(\mathbf{U}_{all}) = \max(\mathbf{U}_{all}) - \min(\mathbf{U}_{all})$. On the other hand, the local diameter is the maximum difference between the clock values in multiset $\mathbf{V}_i$ of vehicle v_i, i.e., $\delta(\mathbf{V}_i) = \max(\mathbf{V}_i) - \min(\mathbf{V}_i)$. As was indicated before, the network topology in VANET is likely to change due to the mobility of vehicles. Therefore, the entities defined above, such as edge set, $\mathbf{U}_{all}$, and $\mathbf{V}_i$ may change as well.

B. Fault Model

Recall that we assume no vehicle is behaving maliciously, e.g., no OBU is manipulating data of BSMs that are broadcast. The source of the problem is assumed to be the absence or manipulation of GPS signals, that is, GPS timing data is omitted or faulty data is injected externally. However, the OBUs themselves behave correctly, as specified. We will study the impact of two types of GPS related faults on clock synchronization in VANET.

The first fault considered is due to GPS signal outage caused by physical obstructions, such as buildings, tunnels, or mountains. In the absence of GPS signals, such outages can cause receivers to have different clock values. This will lead to inconsistencies of information sent in BSMs between vehicles. We observed such faults while conducting lab experiments using Arada OBUs during GPS signal outages. As stated before, in one instance, we observed a difference of 2 seconds between the clock values of two OBUs. A probable cause of OBUs to have different time values is the power-up initialization in the absence of GPS signals.

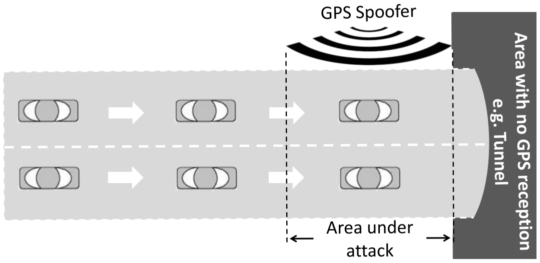

Fig. 3. GPS spoofing attack scenario

The second fault can be induced by GPS time spoofing attacks. The attacker is assumed to be able to spoof GPS signals in a limited geographical area as described in [3]. Attacks can be conducted near areas that suffer from GPS signal outages, like tunnel entrances, or at the exit of parking garages, as shown in Figure 3. Manipulating the time component in the GPS signal allows an attacker to change a GPS receiver's local time.

The impact of GPS time manipulation will be described based on the scenario shown in Figure 1. Due to the hazard, the braking RV emits BSMs containing a hard-braking event during time period T_{brake}. Both faults described above could cause clock differences of more than 500ms between the HV and RV. Recall that the BSM time-to-live should be no more than 500 ms, as stated in [12]. Such time difference would cause the HV to discard BSMs received from the RV, as it would consider these BSMs to be outdated. This could seriously affect the EEBL, as important BSM events may be discarded. The EEBL executing in the HV will fail only if it discards all BSMs containing the event from the RV up to time t_{react}, which is the latest time for the HV's driver to react [18].

V. Clock Synchronization Protocol

The proposed clock synchronization protocol aims to synchronize the clocks of vehicles without having any information about the VANET communication topology. It does not require any extra hardware, nor changes to any standards. Figure 4 outlines the protocol, which is executed on each vehicle's OBU. It will be described below from the viewpoint of the HV. The protocol is divided into two main stages, a *Normal Operation Stage* and an *Agreement Stage*.

A. Normal Operation Stage

The Normal Operation Stage initiates the Agreement Stage just before sending a new BSM. This allows the HV to populate the BSM to be broadcast with the agreement clock value based on the freshest information available. The agreement stage is initiated Δ_o time units before the BSM transmission interval $Timer$ expires, where Δ_o is the projected overhead of the Agreement Stage computations.

Upon receiving a BSM from a neighbor v_j, the receiving vehicle extracts the BSM content, which includes v_j's status

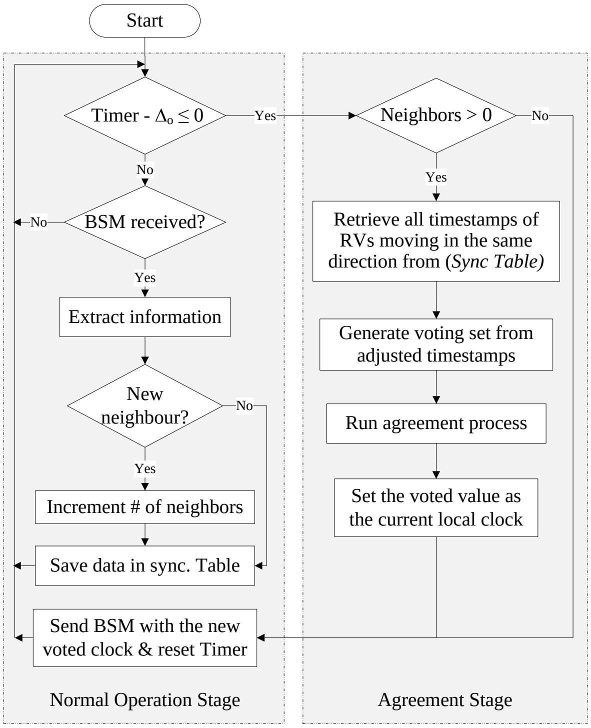
Start
Timer - $\Delta_o \leq 0$
Yes
No
BSM received?
No
Yes
Extract information
New neighbour?
No
Yes
Increment # of neighbors
Save data in sync. Table
Send BSM with the new voted clock & reset Timer
Normal Operation Stage
Neighbors > 0
No
Yes
Retrieve all timestamps of RVs moving in the same direction from (Sync Table)
Generate voting set from adjusted timestamps
Run agreement process
Set the voted value as the current local clock
Agreement Stage

Fig. 4. Proposed clock synchronization protocol

information, its unique vehicle identification (VID), and a *timestamp*. This timestamp is the clock value used by v_j as the result of its agreement algorithm. Each vehicle maintains a table, called *Sync Table*, in which it keeps a record for each neighbor v_j, heading in the same direction, as indicated in the BSM heading field. The record maintains the following information from a neighbor v_j's latest BSM: 1) VID of v_j, 2) the receiving vehicle's local clock value, $t_{rec(j)}$, when the BSM from v_j was received, and 3) the BSM *timestamp*, $t_{RV(j)}$, and 4) the heading of v_j. It should be noted that the heading is saved in case the receiving vehicle changes direction.

If the received BSM is coming from a new neighbor, a new record is created in *Sync Table*, and the neighbor counter is incremented by one. If the BSM comes from an existing neighbor, based on the sender's VID, the data in the existing record will be updated with the latest information received. If no BSM has been received from a vehicle v_j for a predetermined time interval, the record of that vehicle is deleted and the neighborhood count is decremented.

B. Agreement Stage

The Agreement Stage is shown in the right part of Figure 4. If a vehicle has no neighbors, there is no need for agreement and the stage returns control to the Normal Operation Stage, where the BSM will be sent. Otherwise the vehicle retrieves the timestamps $t_{RV(j)}$ of those vehicles with the same heading as its own from the *Sync Table*. The vehicle then generates its sorted voting multiset $\mathbf{V}_i$ based on $t_{k(j)}$ values for each neighbor v_j as described in Subsection IV-A to account for the time elapsed on the vehicle running the agreement process. To obtain the new voted value, the convergence function in Equation (1) is used [7], [22]. The convergence function

$$F_C(\mathbf{V}_i) = Mean[Sel_\sigma(Red^\tau(\mathbf{V}_i))] \tag{1}$$

involves reduction, selection, and computing the mean. The reduction function Red^τ removes the τ largest and smallest elements from multiset $\mathbf{V}_i$. The reduction is not used to account for faulty nodes, as in [22], but to reduce the impact of clock values of vehicles joining a synchronized cluster. Otherwise, each vehicle joining the cluster could trigger the synchronization process.

The selection function Sel_σ selects a submultiset of σ elements from the reduced multiset. Several selection functions are considered: 1) *Fault-Tolerant Midpoint* (FTM), which selects the smallest and largest elements for the mean, 2) *Fault-Tolerant Average* (FTA), which selects all elements for the mean, and 3) *Midpoint*, which selects the median element.

If the diameter $\delta(\mathbf{U}_{all})$ is reduced after each single voting round, the protocol is called *single-step convergent* [22]. This allows clock convergence to occur after a finite number of rounds. At the end of each agreement round, the vehicle sets its local clock to the new voted clock value and sends it in the new BSM. Vehicles in the transmission range receive this BSM and will use this clock value in the voting process of their next round.

As the proposed clock synchronization is an augmentation to the GPS synchronization, the agreement stage is assumed to be always running.

VI. Simulation and Analysis

The following presentation of results demonstrates the impact of diverse parameters on clock convergence using the simplified pathological VANET scenario described above.

A. Simulation Assumptions and Parameters

The simulations, using the Network Simulator Version 3 (NS-3), considered six scenarios, in which N vehicles are moving on a two lane road, such as shown in Figure 3. Vehicles are divided into two fully connected clusters moving in the same direction and with identical constant speed. Each cluster consists of $N/2$ vehicles, where the distance between any two successive vehicles is the same. In the following discussion, we will refer to the graphs of a simple network shown in Figure 2. However, the simulations conducted use a larger number of nodes.

In the simulation, Scenario 1 represents a graph similar to Figure 2b, where the two clusters are disjoint, as the distance $d_{CL(1,2)}$ between the clusters is larger than the transmission range R of 300m. Scenario 2 is similar to Figure 2c, where the first node of the left cluster is connected to the last node of the right cluster. This case of 5-node clusters can be generalized to any number of vehicles by having each vertex in the graph represent one fifth of the cluster nodes. Thus, Scenario 2 can be generalized as a scenario in which the first fifth of the vehicles in the left cluster is connected to the last fifth of the right cluster. The other scenarios follow the same logic. Thus, in Scenario k the first fifth of the left cluster is connected to the $k-1$ fifth vehicles in the right cluster; the second fifth on the left to $k-2$ fifth in the right, and so forth.

The clock values of vehicles of each cluster are initialized with values consistent with the spoofing attack shown in Figure 3, where each vehicle in the area under attack is initialized with the same value. The time difference between the two clusters is initially 30 seconds. This represents half of the maximum value of the timestamp field in a BSM, and is thus the worst-case offset possible. Within each cluster, initial values are random and uniformly distributed from a time interval of 10 seconds. The predefined tolerance Δ_{tol} was chosen to be less than 500ms, which is the *time-to-live* limit in [12]. Whereas a Δ_{tol} of 500ms may seem large for clock synchronization, recall that our goal is not tight synchronization, but fast reaction to events. Thus the EEBL can react to BSMs under such tolerance.

In order to study the influence of traffic density on the clock convergence rate, each of the six scenarios described above are repeated for three different traffic densities. *Low density* assumed $N = 20$ vehicles with a speed of 70 mph and a distance $d_{i,j}$ between successive vehicles v_i and v_j of 60m. Similarly, *medium density* assumed $N = 40$ with a speed of 35 mph and $d_{i,j} = 30$m, and *high density* assumed $N = 80$ with a speed of 20 mph and $d_{i,j} = 15$m. It should be noted that the parameters in the scenarios are carefully chosen to result in the same overall simulation distance. The above parameters represent a traffic flow of 3600 vehicles per hour in each scenario.

To study the impact of reduction and selection functions on the convergence shown in Equation (1), different values for τ and σ are used. Specifically, $Red^{\tau}(\mathbf{V}_i)$, which reduces the τ smallest and largest values of $\mathbf{V}_i$, is simulated for different τ representing reductions of 0, 10%, 20%, and 30%. For each, the four different selection functions Sel_{σ} described in Section V are used. The simulation of each scenario investigated 1) how the global diameter $\delta(\mathbf{U}_{all})$ changed after each round, and 2) the number of rounds taken by each individual vehicle until its local diameter $\delta(\mathbf{V}_i)$ converged to the predefined tolerance Δ_{tol}. Each simulation ended when the local diameter $\delta(\mathbf{V}_i)$ of each vehicle became less than Δ_{tol} or when the number of rounds exceeded 200. The latter was only the case for the non-convergent Midpoint selection functions, as described later.

Each simulation was repeated 100 times and results shown are based on the worst results observed. It is emphasized that the focus of the simulations was on the pathological (worst case) scenarios, not on the best or average behavior.

B. Analysis

During analysis, we observed that the number of rounds to reach agreement used by different scenarios was very similar for reductions of 0, 10%, 20% and 30%. However, the behavior shifted, as will be described. Convergence for reductions of 10%, 20% and 30% was single step, whereas $\tau = 0$ showed oscillations during convergence. Due to space restrictions, we mainly focus on the graphs for 30% in Figure 5, which shows the number of rounds for different scenarios and selection functions. Recall that higher numbered scenarios imply higher degree of cluster overlap, as described in Subsection VI-A. Furthermore, note that resynchronization would be avoided within a cluster if the values of joining vehicles are eliminated as the result of reduction, in which case only the joining nodes would be synchronized.

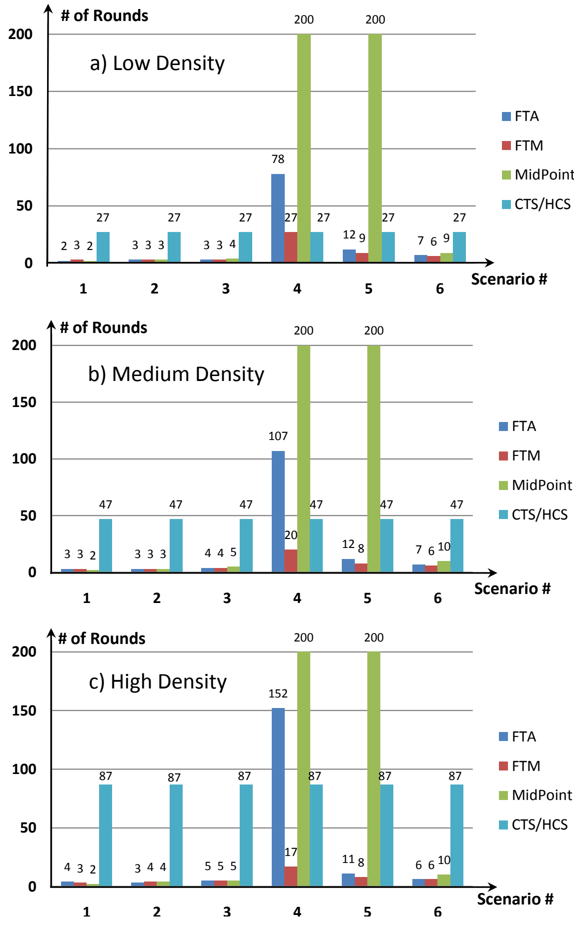

Fig. 5. Impact of selection functions for 30% reduction

The worst case scenario is when only one new value from a different cluster remains in the reduced multiset, because having only one value from the other cluster prolongs synchronization. Figure 5 shows this worst case in Scenario 4 for the three different densities. An example is the case in Figure 2c, with no reduction, or Figure 2d, with one reduced value. The same worst case behavior was observed for lower reduction percentages. For example, this shifting behavior can be seen if one compares reduction of 30% in Figure 5c with that of 20% in Figure 6. In the prior the worst was in Scenario 4, whereas in the latter it was in Scenario 3.

With respect to the selection functions in Figure 5, it is immediately evident that MidPoint shows the worst conver-

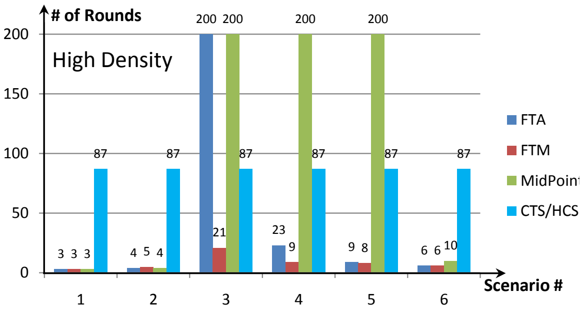

Fig. 6. Impact of selection functions for 20% reduction

gence in Scenarios 4 and 5. In fact it does not converge at all in these scenarios, where the synchronization process was terminated after 200 rounds. In the first three scenarios, the FTA, FTM and MidPoint selections performed nearly identical and outperformed the CTS and HCS. However, in the worst cases Scenarios 4 and Scenario 5, FTM has best performance. This is due to the fact that FTM puts higher weight on the values of the vehicles in the neighbor cluster, thereby speeding up the clock convergence. The scenarios using low, medium and high densities only differ in the number of rounds and FTM performs better as densities increase. Whereas the performance of MidPoint is rather independent of the densities, the FTA used more rounds in denser vehicle distributions. The same was observed for CTS and HCS in all scenarios. The simulations of Figure 6 also showed that MidPoint did not converge. It should be noted that FTA did converge after 517 rounds when the limit on the reduction function was increased.

Based on the results shown, and under consideration of the lower reduction scenarios (i.e. 0 and 10%), which are not shown due to space limitations of this paper, we conclude that the FTM selection function is by far best suited for DSRC safety applications subjected to spoofing attacks in such scenarios.

VII. Conclusion

This research investigated clock synchronization in VANET. A decentralized clock synchronization protocol capable of mitigating against GPS spoofing attacks was presented. The new protocol is based on approximate agreement and it does not require any extra hardware or message overhead. It was compared against the existing protocols CTS and HCS. Within the proposed protocol, the impact of several selection functions on the convergence rate was also investigated. Simulations using Network Simulator Version 3 (NS-3) showed that the new protocol performs best when using the Fault Tolerant Midpoint selection function, especially in worst case scenarios.

In general, this study introduced a theoretical network model and its benefits to DSRC safety applications. The research used simple pathological traffic scenarios that exposed the key issues in GPS time spoofing.

References

[1] Kenney, J. B., *Dedicated short-range communications (DSRC) standards in the United States*, Proc. of the IEEE, vol. 99, pp. 1162-1182, 2011.

[2] S.Wang, A. Pervez, M. Nekovee, *Converging time synchronization algorithm for highly dynamic vehicular ad hoc networks (VANETs)*, 7th International Symposium on Communication Systems Networks and Digital Signal Processing (CSNDSP), 2010.

[3] T. E. Humphreys, B. M. Ledvina, M. L. Psiaki, B. W. OHanlon, and P. M. Kintner, Jr., *Assessing the spoofing threat: development of a portable GPS civilian spoofer*, in Proceedings of the ION GNSS Meeting. Savannah, GA: Institute of Navigation, 2008.

[4] M. Pease, T. Shostak, L. Lamport, *Reaching Agreement in the Presence of Faults*, Journal of the ACM, (1980), 27(2), 228-234.

[5] P. Thambidurai, Y-K, Park, *Interactive Consistence with Multiple Failure Modes*, 7th Reliable Distributed Systems Symposium, (1988), 93-100.

[6] Dolev, D., N.A. Lynch, S.S. Pinter, E.W. Stark, andW.E.Weihl, *Reaching Approximate Agreement in the Presence of Faults*, Proc. Third Symp. on Reliability in Distributed Software and Database Systems, Oct 1983.

[7] Dolev, D., N.A. Lynch, S.S. Pinter, E.W. Stark, andW.E.Weihl, *Reaching Approximate Agreement in the Presence of Faults*, JACM, V. 33, No. 3, pp. 499-516, Jul 1986.

[8] *Standard Specification for Telecommunications and Information Exchange Between Roadside and Vehicle Systems - 5 GHz Band Dedicated Short Range Communications (DSRC) Medium Access Control (MAC) and Physical Layer (PHY) Spec.*, ASTM E2213-03, 2010.

[9] *Amendment of the Commission's Rules Regarding Dedicated Short-Range Communication Services in the 5.850-5.925 GHz Band (5.9 GHz Band)*, Federal Communications Commission FCC 03-324, 2004.

[10] *Vehicle Safety Communications-Applications (VSC-A) Final Report*, DOT HS 811 492 A. U.S. DoT, NHTSA. September 2011.

[11] M. S. Mohamed, S. Hussein and A. Krings, *An Enhanced Voting Algorithm for Hybrid Jamming attacks in VANET*, 2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, 2017, pp. 1-7.

[12] X. Ma, X. Yin, and K.S. Trivedi, *On the Reliability of Safety Applications in VANETs*, Invited paper, International Journal of Performability Engineering Special Issue on Dependability of Wireless Systems and Networks, 8(2), March 2012.

[13] Arada Systems, *www.aradasystems.com*

[14] R. Scopigno and H. A. Cozzetti, *GNSS Synchronization in Vanets*, 3rd International Conference on New Technologies, Mobility and Security, Cairo, 2009, pp. 1-5.

[15] C. Li, Y. Wang and M. Hurfin, *Clock Synchronization in Mobile Ad Hoc Networks Based on an Iterative Approximate Byzantine Consensus Protocol*, 2014 IEEE 28th International Conference on Advanced Information Networking and Applications, Victoria, BC, 2014, pp. 210-217.

[16] J. Elson, L. Girod and D. Estrin, *Fine-grained time synchronization using reference broadcasts*, the Fifth Symposium on Operating Systems Design and Implementation, pp. 147-163, (2002).

[17] L. Li, Y. Liu, H. Yang, H. Wang, *A Precision Adaptive Average Time Synchronization Protocol in Wireless Sensor Networks*, in: Proceedings of the IEEE International Conference on Information and Automation, Zhangjiajie, China, 2008.

[18] Ahmed Serageldin, Hani Alturkostani, and Axel Krings, *On the Reliability of DSRC Safety Applications: A Case of Jamming*, in Proc. International Conference on Connected Vehicles & Expo, ICCVE 2013, Dec. 2-6, 2013, Las Vegas, 2013.

[19] D. Sam and C. Raj, *A time synchronized Vehicular Ad Hoc Network (HVANET) of roadside sensors and vehicles for safe driving*, Journal of Computer Science, vol. 10, no. 10, pp. 1617-1627.

[20] K. Medani , M. Aliouat and Z. Aliouat, *High Velocity Aware Clocks Synchronization Approach in Vehicular Ad Hoc Networks*, Springer International Publishing, 978-3-319-19578-0, pp. 479-490.

[21] Reza Khoshdelniat, Moh Lim Sim, Hong Tat Ewe, and Tan Su Wei, *Time Table Transfer Time Synchronization in Mobile Wireless Sensor Networks*, vol. 5, PIERS Proceedings, Beijing 2009.

[22] Satish M. Srinivasan, Azad H. Azadmanesh, *Data aggregation in partially connected networks*, Computer Communications, Volume 32, March 2009, Pages 594-601