Confirmed-Location Group Membership for Intrusion-Resilient Cooperative Maneuvers*

Júlio Mendonça, Azin Bayrami Asl, Federico Lucchetti[1], Marcus Völp

Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg, Luxembourg

[1]Universität Laussanne, Lausanne, Switzerland

julio.mendonca@uni.lu, azin.bayramiasl@uni.lu, [1]federico.lucchetti@unil.ch, marcus.voelp@uni.lu

***Abstract*—Cooperation among autonomous vehicles is required whenever efficiency or safety prevents maneuvering based solely on the information of individuals. Intersection crossing is a prominent example of such a situation, where obstructed views create safety concerns and where driving on sight would lead to known inefficient solutions. However, communication, a prerequisite for cooperation, and, in general, the complexity of autonomous driving stacks elevate the threat surface beyond justifiable thresholds, creating the potential for cyberattacks to succeed, particularly when targeting the "brain". Some of these attacks go undetected and may harm passengers, pedestrians, and other traffic participants in a vehicle's proximity. In this paper, we address a fundamental challenge of intrusion-resilient maneuver planning: the question of forming consensus groups given variations in the number N of vehicles that participate in complex maneuvers and given that in a larger group of cars, a larger number F may have already been compromised by an adversary. Introducing confirmed-location-based group membership, we show how trust-anchor-provided precise location information can be leveraged to establish a ground truth about N and F to efficiently solve and agree upon intersection crossing as representative of other complex maneuvers in an F fault-and-intrusion tolerant manner.**

***Index Terms*—cooperative driving, fault and intrusion tolerance, group membership, intersection crossing, location**

*This is a pre-print version. The final version will appear in the proceedings of the 99th IEEE Vehicular Technology Conference (VTC-Spring 2024).

I. Introduction

Recently, intrusion resilience by means of trust anchors for detection and recovery [1] and by means of replication and consensus for masking faults [2] has gained renewed attention due to, in part, the increased computational capabilities for autonomous and collaborative driving. However, internal resilience, while necessary to establish trust in increasingly complex functionality, is not enough if the vehicle's individual information and ability to act in isolation are insufficient to perform maneuvers safely and efficiently. For example, when crossing an intersection, rarely more than the lead vehicle in each approaching lane has a full view of the scenario, and often only when it slows down significantly or stops before entering the intersection. This leaves much of the potential of scheduling traffic at slightly larger scales unused (e.g., by scheduling intersection crossing for multiple cars at once). In this example, intersection crossing is only a use case for many traffic situations and locations where complex maneuvers must be executed in a coordinated manner to fall back to unjustifiably slow or, worse, potentially unsafe actions.

Fortunately, intrusion tolerance techniques, such as the replication [3], [4], rejuvenation [5], [6] and diversification [7], [8] triplet, have originally been developed for distributed systems and are thus well prepared to secure fault and intrusion tolerance and resilience across multiple vehicles. This particular triplet allows masking arbitrary behavior of a minority of accidentally faulty or compromised units behind a healthy majority by reaching consensus, operating over extended periods of time by reestablishing the healthy majority (that would otherwise be exhausted), and preventing adversaries from accelerating attacks by presenting it a moving target. Without diversification, adversaries could leverage knowledge gathered on how previous attacks succeeded.

In this work, we shall consider the case where the "brain", though not the driving subsystem of vehicles, might be compromised. Such vehicles may propose arbitrary maneuvers, including unfair (e.g., granting priority to the compromised vehicle) and unsafe ones. We shall leverage replication and consensus to tolerate and safely operate through this behavior. It should also be noted that while modern computing platforms for autonomous driving allow replicating some sensing and driving functionality, applying the same for the entire stack to reach consensus among diverse variants comes at high costs. Our goal is to mitigate the need to replicate such complex functionality by leveraging still correct "brains" in other vehicles that participate in the maneuver.

However, to reach consensus, existing solutions, such as Damysus [9], either assume a fixed number of replicas N, out of which up to F may fail in an arbitrary manner or group membership protocols [10], [11] must be in place to adjust the consensus group by reaching consensus on which nodes to add or remove. Therefore, neither is well suited for driving scenarios, such as intersection crossing, because: (1) traffic changes and, with that, the number of participating vehicles, causing N and hence F to vary over time. Maximizing F to the worst-case scenario that may happen requires a minimum number of vehicles that must always be present to reach an agreement; (2) there may well be phases where no vehicle remains at the maneuver site, which makes it impossible to hand over group membership; (3) infrastructure, such as roadside units (RSUs), despite being expensive to deploy and maintain, faces the same threats, if not more, and can therefore be considered additional replicas. Physical exposure further widens the threat plane of such units; and (4) these solutions cannot determine the ground truth regarding the number and location of vehicles in the absence of infrastructure (e.g., RSUs), making them vulnerable to Sybil attacks.

In this paper, we focus on the problem of efficiently defining group membership for a consensus protocol in cooperative driving. We introduce a novel approach that relies on trust anchors to reliably report a vehicle's location and the space it will occupy in the physical world. We propose a protocol for constrained-location-based group membership and agreement on intersection crossing schedules, solving the previously mentioned critical points that current solutions face. Although our protocol relies on trust anchors, it can be operated in a zero-trust manner after confirming through remote attestation that anchors can, in fact, be trusted. This prevents Sybil attacks because trust anchors provide reliable location information, and vehicles are considered faulty if their location is too close together. We evaluate our solution using the MOSAIC framework and the SUMO traffic simulator.

We introduce in Sec. III our system model and trust-anchor requirements. Sec. IV presents our confirmed-location group-membership protocol for forming consensus groups. Sec. V detail how the protocol works in an intersection scenario. Sec. VI presents the results from our experimental evaluation. Sec. VII concludes the

paper and highlights future work.

II. Related Work

In the upcoming era of autonomous self-driving vehicles, effective coordination becomes crucial for managing shared resources like intersections and parking spaces. Integrating these vehicles aims to reduce traffic and pollution through optimized routes [12]. However, ensuring safe circulation requires solutions for coordinating their activities and movements. The identified vital solutions highlight varying levels of autonomy in decision-making during the coordination process, emphasizing the need for careful consideration [13].

Intersections pose significant challenges in traffic coordination and contribute to a notable portion of traffic accidents [14]. Autonomous Intersection Management (AIM) enhances traffic flow compared to traffic signals, minimizing congestion [15]. Connected and Automated Vehicles (CAVs) may leverage their connectivity to create Cooperative Intersection Management (CIM), facilitating negotiation at signal-equipped or non-equipped intersections [14]. For instance, Yosodipuro et al. [15] introduced TLRRIM and VESC for managing mixed-traffic intersections, focusing on a traffic-load-responsive reservation and V2X-enabled speed coordination to improve traffic flows.

Many works have investigated intersection crossing as an example of a complex driving maneuver [16], [17], including how to optimally schedule traffic to minimize time-to-cross or consumed energy [18]. Xu et al. [19] proposed a fixed N and F consensus protocol for highway joining. Liu et al. [20] propose using a challenge-based approach to form a consensus group among vehicles to decide on traffic reports. They mention using location-based challenges as a possible implementation for their approach. Regnath et al. [21] suggest reaching a consensus among vehicles in the front of approaching lanes of an intersection.

Consensus in distributed systems is widely studied [3], [4], [9], and applied as well locally to protect individual systems [2], [22]. Group membership [10], [11] allows changing the consensus group, even in response to situational changes, such as increasing or decreasing threats [23]. Li et al. [24] explored distributed fault-tolerant consensus algorithms for coordination. However, classical group membership solutions are not well suited for connected vehicle maneuvering, as explained before. We shall, therefore, focus on confirmed-location-based membership.

Our work leverages trust anchors [1] and an intrusion-tolerant systems architecture [2]. We shall argue what functionality must be provided by such a trusted entity to safely and securely operate vehicles through complex maneuvers.

One such functionality is the trust anchor's ability to provide reliable and precise location information, demonstrating a vehicle's location on the road. Obtaining precise location information is not trivial and has been studied in many works [25]. Autonomous vehicles rely on precise position estimation using sensor measurements and signal processing, requiring reliable and accurate localization, especially in the face of potential sensor failures [26]. Aside from military-grade GPS [27], which we assume will not be available for civil autonomous driving, many such solutions either require complex software stacks for perceiving and analyzing the vehicle's environment to more accurately pinpoint it on the map [28], or they are prone to attacks [29], which might be as simple as physically displacing a roadside unit used to augment the GPS signal. Shin [26] presented a system and architecture for fault-tolerant localization on a prototype test vehicle, incorporating differential GPS, an Inertial Navigation System (INS), a camera,

and laser range-finding sensors.

III. Trust Anchor, System, and Fault Model

The ability of adversaries to turn fully compromised vehicles into cyber-kinetic weapons necessitates the inclusion of a subsystem that is trusted not to fail, even if the remainder of the vehicle's hardware and software fall into the hands of adversaries. The fundamental question to which this work contributes is answering what functionality this *trust anchor* must provide for the vehicle to safely operate through attacks, even if they are successful in compromising all functionality outside that anchor. We shall see later that although establishing the trustworthiness of an anchor is an orthogonal question, our use of a random nonce already allows for remotely attesting anchors and their trustworthiness.

For the purpose of this work, we assume the trust anchor provides the necessary functionality to drive the vehicle alongside a given trajectory. This includes basic safety checks, such as brake-on-imminent-collision (e.g., by evaluating front radar). We further assume the trust anchor to have access to precise and reliable location information and to a time-of-day clock that is coarsely synchronized with the clocks of other vehicles up to a precision Π, which we assume to be in the range of $\approx 10s$. The trust anchor may internally be replicated to offer this functionality in a reliable way [2]. Works such as Gouveia et al. [22] and Schulz et al. [30] illustrate how such anchors can be implemented.

In addition to the trust anchor, we assume vehicles are comprised of autonomous driving and a V2X communication stack, which may be too large and resource-hungry to justify replication inside a singular car. Consequently, these stacks may fall prey to adversaries and behave arbitrarily and potentially maliciously when compromised.

The interface to the trust anchor is as follows:

`drive(trajectory, loc, ℂ)`

Instructs the trust anchor to follow the given `trajectory` once the vehicle is in location `loc` and provides the certificate $\mathbb{C}$ to confirm the trustworthiness and freshness of the trajectory and location information. Trust anchors stop vehicles (when possible on a side lane) if they reach a critical location (e.g., the entry of the intersection) without having been passed a valid trajectory. The trust anchor will also anticipate the stopping of vehicles in front and stop the ego vehicle accordingly. Our primary goal is to ensure safety under attack. To resume from such a stop, the following vehicles will, therefore, have to engage in agreeing on a maneuver to safely pass the stopped vehicle (e.g., by recursively applying the solution we sketch for intersection crossing).

In addition to driving, we require the trust anchor to provide the following two functions:

`certifyLocation(m)` $= C\langle id_i, loc_i, t_i, nonce_i, m\rangle\sigma_i$

Signs message m along with the quadruple id_i, loc_i, t_i, $nonce_i$, comprised of the vehicle identifier, its current location, time of day and a random value that is large enough and sufficiently unpredictable to prevent precomputing responses. We call this a *location certificate* for message m and assume loc_i to report both the position and the velocity of the vehicle. Additionally, time of day with precision Π prevents replaying previously collected quadruples since this would require positioning the compromised vehicles at that location to then return within Π to the same location for mounting the attack.

`localize(m)` $= L\langle id_i, loc_i, t_i, m\rangle\sigma_i$

Adds the vehicle's identifier, location, and time of day to a message m, signing it to prove origin from the trust anchor. Opcodes L and C differentiate these two messages to prevent adversaries from obtaining a certified location by passing a chosen nonce as m into `localize`.

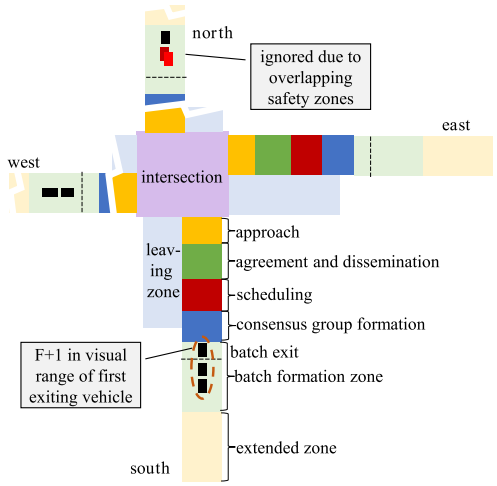

Fig. 1. Zones for zone-based group membership and zonal intersection crossing. Vehicles in the batch formation zone by the time the first unassigned vehicle exits form the set of cars from which the consensus group is formed. Vehicles overlapping in their safety zone are thereby ignored.

Unlike previous solutions, our fault model is dynamic in the number of vehicles present within a certain area of interest (e.g., on the lanes approaching an intersection). To that end, we parameterize our solution by a constant[1] $c \geq 4$, such that if N vehicles are present in a given area, up to $F < \frac{N}{c}$ vehicles may be compromised. In the next section, we shall require $F + 1$ vehicles to be in the visual range of the first car leaving the area of interest and $N \geq c$ vehicles to be in the approaching lanes batch formation zones. We leave the extension of our work to more sparsely populated areas as future work while already supporting empty zones.

IV. Location can Replace Group Membership

We determine consensus group membership by leveraging the location of vehicles in approach to the intersection. Vehicles intersecting in their safe-driving zones and non-responsive vehicles are ignored and will not receive a maneuver certificate $\mathbb{C}$. Their trust anchors will stop them before entering the intersection crossing zone. For the purpose of approaching the maneuver zone, we consider constant and bounded velocities for all cars in a lane.

Fig. 1 shows the batch formation zones for triggering confirmed-location-based group membership and further

```
1  // localization phase
2  nextBatch = #(id_i) | 0;

3  on ⟨closeBatch, ..., b⟩ delivered:
4    if in the extended zone
5      nextBatch = b+1

6  on vehicle id_i seeing unassigned vehicle id_j
7          at loc_j^b leaving batch formation zone:
8    broadcast certifyLocation(
9          ⟨closeBatch, loc_j^b, id_j, ℂ_i, nextBatch⟩)

10 on m = C⟨id_j, loc_j, t_j, nonce_j,
11         ⟨closeBatch, loc_k^b, id_k, b⟩⟩σ_j :
12   set timeout
13   broadcast localize(⟨Intent, dest, m⟩)

14 // batch formation phase
15 on timeout:
16   validate time, location, nonce, and signatures
17      of each m = L⟨id_m, loc_m, t_k, C⟨id_j, loc_j, t_j, nonce_j,
18              ⟨closeBatch, loc_k^b, id_k, b⟩⟩σ_k⟩σ_m
19   group m by id_j and sort vehicles in groups by
20              lane and distance
21   remove vehicles with intersecting safety zones
22   Let L[id_j] be the sorted group for id_j
23   discard L[id_j] if |L[id_j]| < N_min
24   deliver L = L[id_j] where id_j is closest to the
25       batch exiting vehicle break ties deter-
26       ministically (e.g., by lane)
27   N_obs = |L|
28   F = max integer, such that F < (N_obs)/(c-1)
29   CG = {L[1], ... L[3F + 1]}
```

Fig. 2. Localization and group-membership protocol.

[1]We describe our solution for homogeneous consensus protocols, which typically assume a fixed number of replicas $N = 3F+1$ and a quorum of size $Q = 2F+1$. Notice however, that our protocol works equally well for hybrid protocols with $N = 2F + 1$ and $Q = F + 1$. In this case, $c \geq 3$.

areas that the vehicles will pass while advancing with the intersection-crossing protocol. It also shows the extended zone, which we use to handle back pressure. Vehicles not already assigned to a zone execute the group membership protocol depicted in Fig. 2, once they observe a leading car in visual range to leave the batch formation zone (*batch exit*).

While in the extended zone, vehicles learn about formed batches by cars in the previous batch zone (Ln.5). They use this information to extend the batch numbers and to compensate for back pressure. We assume the extended zone is at least as large as the batch zone so that all vehicles learn about batch closures unless the batch formation zone of all approaching lanes is empty. In this case, vehicles revert to the default batch numbers by hashing their ID for uniqueness and by resuming from batch 0 (Ln.2).

Once a vehicle id_i crosses or sees a previous vehicle id_j crossing the batch exit marker (dashed line) that is not already assigned to a batch, it triggers the closure and formation of a new batch (Ln.7). To do so, it broadcasts the location of the crossing vehicle (Ln.9) alongside a location certificate for itself to all cars in all batch formations and extended zones of all approaching lanes.

Receiving such a message (Ln.11) from an observer of an exiting vehicle, including from itself, vehicles program a timeout that is large enough for all vehicles in the batch formation zones to receive all location certificates and to deliver this message. After that, they communicate their intent (i.e., the destination lane $dest$) and reflect the closure information by localizing both using their trust-anchors `localize` function.

Once this timeout fires, a reliable broadcast of the messages ensures that if one healthy vehicle gets delivered the above information, then all healthy cars will eventually be delivered this information, and the timeout ensures that enough time for this delivery has been granted. For that reason, healthy vehicles will process the subsequent steps of the protocol on the same information (even if they received the individual messages in a different order). Upon receiving a localized `Intent` message, they validate the embedded time-of-day timestamps to be within Π, while accounting for communication delays, whether the reported locations are, in fact, on the lane and whether nonces of the `closeBatch` and `Intent` messages match and whether their signatures can be verified (Ln.16). Discarding invalid messages, vehicles then group messages by the id_j of the observer and sort them by lane and by distance from the intersection (Ln.19), removing again vehicles that overlap in their safety zones as this indicates rigged cars with more than one trust anchor (Ln.21). Without this, adversaries mounting a Sybil attack would be able to manipulate the ground truth about N. We also discard sorted sets smaller than the minimum size $N_{min} = c$ that we need to form a consensus group (Ln.23), leaving the handling of sparse but non-empty batches for future work. From the remaining groups, we deliver the one whose observer is closest to the batch exiting vehicle, breaking ties deterministically when vehicles in multiple lanes exit at the same time (e.g., north before east before south before west). The cardinality of this group is the observed number of vehicles N_{obs}. From this, we obtain the maximum number F of faulty vehicles in the batch as the maximum integer that fulfills $F < \frac{N_{obs}}{c-1}$ (Ln.28) and form the consensus group CG as the first $3F+1$ (or $2F+1$) vehicles in that group (depending on whether homogeneous or hybrid consensus is used).

When observing vehicles in the above manner, it may happen that faulty vehicles do not engage in the protocol, either by triggering the closure, which we compensate by all observers in visual range initiating the protocol, or by not responding to their intent. Again, leaving the handling of sparse intersections as future

work, we assume at least $F+1$ observers to be present in the lane that closes the batch. In consequence of possibly not learning about faulty vehicles, N_{obs} may range anywhere in the interval $[N-F, N]$, and because $F < \frac{N}{c}$, we might have missed vehicles indicating a larger F. To compensate for that, we pessimistically assume $N = N_{obs} + F$, which leads to $F < \frac{N_{obs}}{c} + \frac{F}{c}$ and $F < \frac{N_{obs}}{c-1}$ (as in Ln.28). This also explains why $c \geq 4$ ($c \geq 3$), even though consensus can already be reached with $N = 3F+1$ ($N = 2F+1$) vehicles.

Notice also that our solution to certified-location-based group membership is able to leverage any standard, reliable broadcast protocol [31]–[33] to ensure that all healthy vehicles in the batch-formation and extended zone receives information sent by a healthy vehicle (see Ln.9 and 13), even if these protocols assume a fixed N and F, which is only available after localization. We do so by tapping into the state of these protocols to speculatively deliver received messages alongside witness or round information until the timeout (Ln.12) terminates the batch-formation phase for the individual vehicles. Then, for all valid groups, we determine N_{obs} and F (as in Ln.27 and 28) to determine which group can be delivered.

Vehicles move at a constant, bounded velocity in each lane. Therefore, they have completed the algorithm in Fig.2 within a bounded amount of time, during which they continue to approach the intersection. Given a maximum allowed approaching velocity, we can, therefore, reserve a zone large enough to ensure the completion of this consensus group formation algorithm (see Fig. 1).

V. Intersection Crossing

Intersection crossing proceeds by scheduling the crossing of all N_{obs} vehicles in L using a deterministic scheduling algorithm. Provided the result can be verified (e.g., by checking for collisions), the schedule computation can be parallelized by splitting and distributing the task into $k = \left\lfloor \frac{N_{obs}}{F+1} \right\rfloor$ subtasks (e.g., such that the vehicles $L[(k-1)(F+1)], \ldots L[(k)(F+1)]$ execute subtask k). With $F+1$ vehicles per subtask, one healthy vehicle will produce a correct result, which can be verified and selected. Otherwise, if results can not be verified by just analyzing the response (e.g., when aiming for fairness or when optimizing fuel consumption), groups with $2F+1$ vehicles can be formed to select the matching result of $F+1$ cars, which is guaranteed to include the result of a healthy vehicle. While scheduling, vehicles move through the scheduling zone in their lane, communicating results back to the consensus group, which finalizes the F-fault-and-intrusion-tolerant intersection-crossing protocol by reaching an agreement on and disseminating the result. For the former, an arbitrary homogeneous (with $c \geq 4$ or hybrid with $c \geq 3$ and $|CG| = 2F+1$) consensus protocol can be used, and dissemination is again by means of reliable broadcast. The agreement certificate, together with the certificate attesting to the formation of L and hence N_{obs} and F thereby form the information $\mathbb{C}$ that needs to be passed to the trust anchor for validation. Vehicles that do not receive a valid intersection crossing schedule by the time they exit the agreement and dissemination zone will be stopped by their trust anchor before blocking the intersection. We handle backpressure by delaying the execution of the schedule by the time the previous batch (if any) needs to evacuate the intersection and confirm this delay by passing the previous batch's certificate $\mathbb{C}$ as proof.

VI. Evaluation

To evaluate our approach, we have implemented our group membership and agreement protocol in the MOSAIC framework [34]. MOSAIC integrates traffic simulators, such as Sumo [35], and comes with an integrated

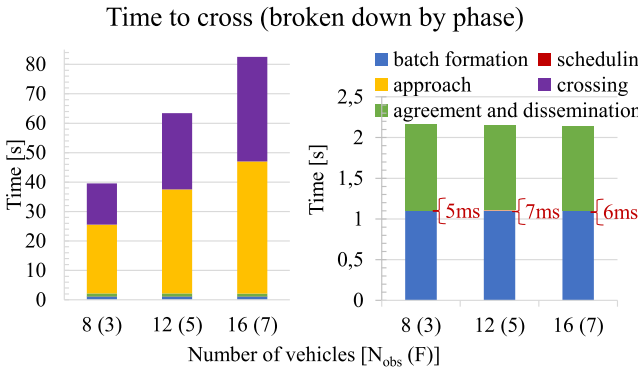

Fig. 3. Time to cross for different numbers of vehicles in the batch formation zone and hence for a different number of tolerated to be faulty vehicles. Times are broken down by phase, and we zoom on the right into the batch formation, scheduling, agreement, and dissemination phases.

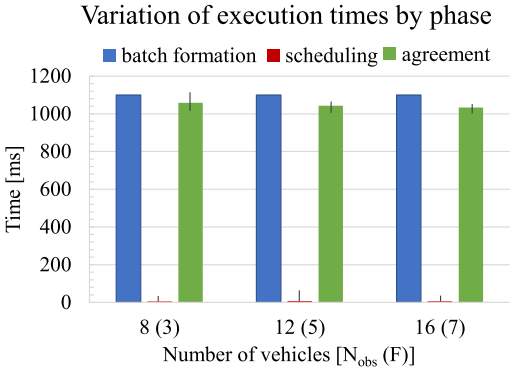

Fig. 4. Zoom in and variation of the execution times of the different phases (in terms of observed best / worst case).

network simulator. We simulate realistic traffic behavior for a four-lane intersection and run the proposed protocol simulating a V2X wireless network following the standards of IEEE 802.11P. This standard offers data rates from 3 to 27 Mbit/s, tailored for vehicular communication [36]. Besides, we pessimistically configured the network with random delays of 10-100 ms, according to the information available in [37], [38]. Furthermore, since our protocol is zone-based, we define the size of the batch formation zone as 50 meters, which allows for up to four vehicles per approaching lane (16 in total).

We measured the time to cross (Figure 3), zooming into the individual phases, their variation (Figure 4), and the consumed overall network bandwidth (Figure 5) for different numbers of vehicles in the batch formation zones and hence different numbers of faults that our protocol is able to tolerate. Measurements were performed on 100 runs in MOSAIC choosing, for reproducibility, with random seeds that are equal to the number of the run (i.e., 1 - 100). Scheduling overheads were measured on an Intel i9-11950H PC, running at 2.60 GHz, with 32GB of RAM, under Windows 10 version 21H2. Since our approach is independent of the actual scheduling algorithm that is used for determining how to cross the intersection and since we used only a first-come-first-served strategy, scheduling overheads are not significant. Still, as can be seen in the next subsections, our approach allows for much costlier execution of scheduling algorithms and supports parallelization into computation groups of $2F+1$ (or, in case schedules can be validated, $F+1$) vehicles.

A. *Time to cross*

Figure 3 shows the time-to-cross, broken down into the individual phases (on the left) and zooms into the batch formation, scheduling, and agreement phases (on the right). As can be seen, the overhead added by the batch formation and the consensus is marginal (less than 15% compared to the time to cross the intersection and less than 6% to approach and cross). We vary vehicle arrival rates from $N_{obs} = 8$ to $N_{obs} = 16$ vehicles, which corresponds to the tolerated number of faulty vehicles from $F = 3$ to $F = 7$ (for $c = 3$, assuming hybrid and/or synchronous consensus).

We have repeated all measurements 100 times, zooming into the overhead of the individual phases (aside from approaching) and showing the observed best and worst

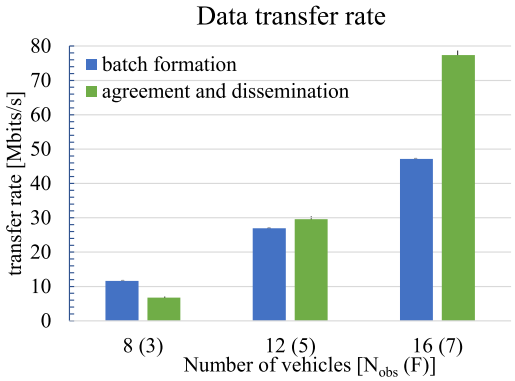

Fig. 5. Data transfer rates for all vehicles during the two phases in which communication happens: during batch formation and agreement and dissemination.

case execution times in Figure 4. Variations in communication times are insignificant (below 6% relative to the shown average) in the integrated simulator. However, we expect more elaborate results from repeated measurements with more realistic network simulators (such as OMNeT++ [39]) and when complicating the intersection scenario (e.g., to multiple lanes per direction).

B. Data transfer rates

Figure 5 shows the overall data transfer rates needed by our protocol during the two communication phases (i.e., during batch formation and during agreement). As can be seen, rates grow linearly in the number of vehicles times the number of faults for batch formation since $F+1$ vehicles broadcast closure and N vehicles respond by broadcasting their intent. Consensus, on the other hand, grows quadratic due to the used consensus algorithm, broadcasting $|CG|$ messages to all cars of the batch and the following batch. The bandwidth required for individual vehicles is approximately $\frac{1}{N_{obs}}$ the reported number.

C. Impact of the constant c in the protocol

Since our proposed protocol can be parametrized to work with different configurations, we measured the impact that a chosen c has over the data transfer rater of the consensus phase. By changing the value of c the system can be adapted to tolerate more or less rogue vehicles, meaning that bigger c implies a smaller number of tolerated rogue vehicles F.

Considering the same number of vehicles in a batch (i.e., $N_{obs} = 16$), c does not impact messages exchange during batch formation, as all the cars need to communicate equally to be part of a batch. However, c impacts the consensus group size and, therefore, the number of messages to agree on a computed schedule. Table I shows the overall data transfer rate used by all vehicles in the batch during the agreement and dissemination phase of the protocol. The results clearly show the trade-off between the data transfer rate and the number of tolerated faulty or rogue vehicles. With $c = 3(F = 7)$, the system would be more resilient against faulty or rogue cars, but it comes with a network overhead. In crowded scenarios, this configuration may not be efficient since it could saturate the available network. On the other hand, choosing $c = 4(F = 5)$ significantly reduces the data transfer rate while preserving a reasonable number of tolerable rogue and faulty vehicles. The configurations with $c = 6(F = 3)$ and $c = 9(F = 1)$ are more lightweight, and they would be of good use in scenarios where the available network bandwidth is unsuitable for a high number of message exchanges.

To summarize, our protocol delivers fault and intrusion tolerance for cooperative driving without the need for fixed infrastructure. Compared to a non-fault-tolerant solution, it achieves this without a significant performance penalty. Our protocol can adapt to tolerating varying number of faults or rogue vehicles, depending on how many vehicles are present in the batch formation zone by the time a previously unassigned vehicle

TABLE I

OVERALL DATA TRANSFER RATE FOR N VEHICLES DURING THE AGREEMENT AND DISSEMINATION PHASE, CONSIDERING THE CHOSEN c FOR THE PROTOCOL.

Overall data transfer rate (Mbits/s)			
Configuration	Average	Min	Max
N=16, C=3 (F=7)	77.3644	75.32063	78.73396
N=16, C=4 (F=5)	29.51153	28.47576	30.20559
N=16, C=6 (F=3)	8.161355	7.587071	8.461247
N=16, C=9 (F=1)	0.569509	0.389318	0.676919

exits such a zone. This improves over the state-of-the-art, which relies either on membership protocols and the presence of vehicles that can reach consensus to negotiate who joins or leaves the consensus group or that depend on infrastructure (e.g., RSUs) to serve as members in case too few vehicles are present to reach consensus. The need to reach consensus for each vehicle joining or leaving comes with a significant performance penalty, which our protocol avoids.

VII. CONCLUSIONS

Introducing trajectory following and precise localization as trusted functionalities, we demonstrated in this paper how group membership can be based on the space vehicles assumed in the physical world. Constraining valid vehicles to one sender per location, we solve group membership for an initially unknown number of cars N and hence for an initially unknown fault threshold F. We utilize this group to tolerate intrusions during the schedule computation and when agreeing on which schedule to apply for crossing the intersection. Our Sumo-based evaluation shows that our solution offers fault- and intrusion resilience at a marginal cost on the time to approach and cross the intersection (less than 15% overhead of the time needed to cross and 6% compared to approach and cross), while we can tolerate a number F of faulty vehicles that change with the number N of vehicles that approach the intersection.

Future work includes addressing sparsity and the complexity of reliable and precise localization and the extension of our approach to maneuvers (such as takeover) that are not bound to a location. In such maneuvers, zones can only be defined once agreement about performing the maneuver is reached.

ACKNOWLEDGMENT

This work has been supported by the Luxembourg Fond Nationale de Recherche (FNR) and the German Research Council (DFG) through the CORE Inter Projects ByzRT (C19-IS-13691843) and ReSAC (C21/IS/15741419).

REFERENCES

[1] J. Han and A. Cho, "Practical in-vehicle security architecture based on trust anchors," in *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)*, 2023, pp. 1–3.

[2] A. Shoker, V. Rahli, J. Decouchant, and P. Esteves-Verissimo, "Intrusion resilience systems for modern vehicles," in *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)*, 2023, pp. 1–7.

[3] M. Castro and B. Liskov, "Practical byzantine fault tolerance and proactive recovery," *ACM Trans. Comput. Syst.*, vol. 20, no. 4, p. 398–461, nov 2002. [Online]. Available: https://doi.org/10.1145/571637.571640

[4] P. Esteves-Veríssimo, N. F. Neves, and M. P. Correia, "Intrusion-tolerant architectures: Concepts and design," *Architecting Dependable Systems. Lecture Notes in Computer Science*, vol. 2677, 2003.

[5] P. Sousa, N. F. Neves, and P. Esteves-Verissimo, "Proactive resilience through architectural hybridization," in *Proceedings of the 2006 ACM symposium on Applied computing*. ACM, 2006, pp. 686–690.

[6] P. Sousa, A. N. Bessani, M. Correia, N. F. Neves, and P. Esteves-Verissimo, "Highly available intrusion-tolerant services with proactive-reactive recovery," *IEEE Trans. Parallel Distrib. Syst.*, vol. 21, no. 4, pp. 452–465, Apr. 2010. [Online]. Available: http://dx.doi.org/10.1109/TPDS.2009.83

[7] B. Spengler, "Pax: The guaranteed end of arbitrary code execution," grsecurity.net avail at. https://grsecurity.net/PaX-presentation.pdf, Oct. 2003.

[8] T. Roeder and F. B. Schneider, "Proactive obfuscation," *ACM Transactions on Computer Systems (TOCS)*, vol. 28, no. 2, July 2010.

[9] J. Decouchant, D. Kozhaya, V. Rahli, and J. Yu, "DAMYSUS: streamlined BFT consensus leveraging trusted components," in *Proceedings of the 17th European Conference on Computer Systems*, 2022, pp. 1–16.

[10] M. K. Reiter, "A secure group membership protocol," *IEEE Transactions on Software Engineering*, vol. 22, no. 1, pp. 31–42, 1996.

[11] A. Das, I. Gupta, and A. Motivala, "Swim: Scalable weakly-consistent infection-style process group membership protocol," in *DSN*, 02 2002, pp. 303–312.

[12] N. Menon, N. Barbour, Y. Zhang, A. R. Pinjari, and F. Mannering, "Shared autonomous vehicles and their potential impacts on household vehicle ownership: An exploratory empirical assessment," *International Journal of Sustainable Transportation*, vol. 13, no. 2, pp. 111–122, 2019.

[13] S. Mariani, G. Cabri, and F. Zambonelli, "Coordination of autonomous vehicles: taxonomy and survey," *ACM Computing Surveys (CSUR)*, vol. 54, no. 1, pp. 1–33, 2021.

[14] L. Chen and C. Englund, "Cooperative intersection management: A survey," *IEEE transactions on intelligent transportation systems*, vol. 17, no. 2, pp. 570–586, 2015.

[15] N. D. Yosodipuro, E. Javanmardi, J. Nakazato, Y. Tamura, X. Défago, and M. Tsukada, "Mixed-traffic intersection management using traffic-load-responsive reservation and v2x-enabled speed coordination."

[16] A. Rausch, N. Oswald, and P. Levi, "Cooperative crossing of traffic intersections in a distributed robot system," in *Sensor Fusion and Networked Robotics VIII*, P. S. Schenker and G. T. McKee, Eds., vol. 2589, International Society for Optics and Photonics. SPIE, 1995, pp. 218 – 229. [Online]. Available: https://doi.org/10.1117/12.220960

[17] S. A. Fayazi and A. Vahidi, "Mixed-integer linear programming for optimal scheduling of autonomous vehicle intersection crossing," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 3, pp. 287–299, 2018.

[18] M. Amouzadi, M. O. Orisatoki, and A. M. Dizqah, "Optimal lane-free crossing of cavs through intersections," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 2, pp. 1488–1500, 2023.

[19] W. Xu, A. Willecke, M. Wegner, L. Wolf, and R. Kapitza, "Autonomous maneuver coordination via vehicular communication," in *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, 2019, pp. 70–77.

[20] H. Liu, C.-W. Lin, E. Kang, S. Shiraishi, and D. M. Blough, "A byzantine-tolerant distributed consensus algorithm for connected vehicles using proof-of-eligibility," in *Proceedings of the 22nd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, ser. MSWIM '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 225–234.

[21] E. Regnath, M. Birkner, and S. Steinhorst, "Ciscav: Consensus-based intersection scheduling for connected autonomous vehicles," in *IEEE Int. Conf. on Omni-Layer Intelligent Systems (COINS)*, 2021, pp. 1–7.

[22] I. P. Gouveia, M. Völp, and P. E. Veríssimo, "Behind the last line of defense: Surviving soc faults and intrusions," *Comput. Secur.*, vol. 123, p. 102920, 2022. [Online]. Available: https://doi.org/10.1016/j.cose.2022.102920

[23] D. S. Silva, R. Graczyk, J. Decouchant, M. Völp, and P. Esteves-Verissimo, "Threat adaptive byzantine fault tolerant state-machine replication," in *2021 40th International Symposium on Reliable Distributed Systems (SRDS)*, 2021, pp. 78–87.

[24] T. Li, L. Tseng, T. Higuchi, S. Ucar, and O. Altintas, "Poster: Fault-tolerant consensus for connected vehicles: A case study," in *2021 IEEE Vehicular Networking Conference (VNC)*. IEEE, 2021, pp. 133–134.

[25] D. Kumar and N. Muhammad, "A survey on localization for autonomous vehicles," *IEEE Access*, 2023.

[26] B. Shin, "Fault tolerant control and localization for autonomous driving: Systems and architecture," UCB/EECS-2016-83, Tech. Rep., 2016.

[27] W. Mansfeld, *Global Positioning System (GPS)*. Wiesbaden: Vieweg+Teubner Verlag, 1998, pp. 113–215. [Online]. Available: https://doi.org/10.1007/978-3-322-92917-4_3

[28] P. Yao, X. Sui, Y. Liu, and Z. Zhao, "Vision-based environment perception and autonomous obstacle avoidance for unmanned underwater vehicle," *Applied Ocean Research*, vol. 134, p. 103510, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0141118723000512

[29] J. Shen, J. Y. Won, Z. Chen, and Q. A. Chen, "Demo: Attacking multi-sensor fusion based localization in high-level autonomous driving," in *2021 IEEE Security and Privacy Workshops (SPW)*, 2021, pp. 242–242.

[30] S. Schulz, A. Schaller, F. Kohnhäuser, and S. Katzenbeisser, "Boot attestation: Secure remote reporting with off-the-shelf iot sensors," in *Computer Security – ESORICS 2017*, S. N. Foley, D. Gollmann, and E. Snekkenes, Eds. Cham: Springer International Publishing, 2017, pp. 437–455.

[31] D. Dolev, "The byzantine generals strike again," *Journal of Algorithms*, vol. 3, no. 1, pp. 14–30, 1982. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0196677482900049

[32] D. Imbs and M. Raynal, "Simple and efficient reliable broadcast in the presence of byzantine processes," *CoRR*, vol.

abs/1510.06882, 2015. [Online]. Available: http://arxiv.org/abs/1510.06882

[33] G. Bracha and S. Toueg, "Asynchronous consensus and broadcast protocols," *J. ACM*, vol. 32, no. 4, p. 824–840, oct 1985. [Online]. Available: https://doi.org/10.1145/4221.214134

[34] K. Schrab, M. Neubauer, R. Protzmann, I. Radusch, S. Manganiaris, P. Lytrivis, and A. J. Amditis, "Modeling an its management solution for mixed highway traffic with eclipse mosaic," *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[35] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018. [Online]. Available: https://elib.dlr.de/124092/

[36] A. Zekri and W. Jia, "Performance evaluation of rate adaptation algorithms in ieee802.11p heterogeneous vehicular networks," in *2018 IEEE 15th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, 2018, pp. 107–115.

[37] A. Finkenzeller, A. Mathur, J. Lauinger, M. Hamad, and S. Steinhorst, "Simutack-an attack simulation framework for connected and autonomous vehicles," in *97th Vehicular Technology Conference (VTC2023-Spring)*. IEEE, 2023, pp. 1–7.

[38] F. Pollicino, S. Eisa, P. Rosa, M. L. Pardal, and M. Marchetti, "Decentralized position detection for moving vehicles," in *97th Vehicular Technology Conference (VTC2023-Spring)*. IEEE, 2023, pp. 1–6.

[39] S. Unterschütz, A. Weigel, and V. Turau, "Cross-platform protocol development based on omnet++." in *SimuTools*. Citeseer, 2012, pp. 278–282.