A Secure Alert Messaging System for Safe Driving

Wafa Ben Jaballah[a,*], Mauro Conti[b], Mohamed Mosbah[a], Claudio E. Palazzi[b]

[a] *University of Bordeaux, Polytechnic Institute of Bordeaux, LaBRI, CNRS, UMR 5800, France*
[b] *Department of Mathematics, University of Padua, Italy*

Abstract

Vehicular safety is an emergent application in inter-vehicular communications. As this application is based on fast multi-hop message propagation, including information such as position, direction, and speed, it is crucial for the data exchange system of the vehicular application to be resilient to security attacks. To make vehicular networks viable and acceptable to consumers, we have to design secure protocols that satisfy the requirements of the vehicular safety applications. The contribution of this work is threefold. First, we analyze the vulnerabilities of a representative approach named Fast Multi-hop Algorithm (FMBA) to the position cheating attack. Second, we devise a fast and secure inter-vehicular accident warning protocol which is resilient against the position cheating attack. Finally, an exhaustive simulation study shows the impact of the attack on the protocol FMBA on delaying the transmission of alert messages. Furthermore, we show that our secure solution is effective in mitigating the position cheating attack.

Keywords: Inter-Vehicular Communication, Vehicular Safety, Alert Messaging Warning, Position Cheating Attack.

1. Introduction

Annually, road crashes result in almost 120,000 fatalities and 2.4 million injuries in the European Region [1]. Road traffic injuries represent the leading cause of death among adolescents and young adults. Moreover, the economic burden of road crashes is as much as 3% of gross domestic product. Although, many potential preventive strategies exist [2], they are not completely effective. Hence, it is desirable to take up the challenge and reduce the burden of road traffic injuries. The basic approach consists in using advanced technologies that can prevent vehicles from being involved in accidents. In this direction, one of the most promising techniques is based on the use of inter-vehicular communication (IVC) [3]. In fact, many applications are possible in this context, yet local danger warning systems remain the most prominent ones. Clearly, the effectiveness of such safety related application is based on the reliability of the broadcast information.

Alert messaging is a building block component of intelligent transportation systems and an emergent application for vehicular communications. Vehicles can communicate between each other, without needing the intervention of any

*Corresponding Author
Email addresses: wafa.benjaballah@labri.fr (Wafa Ben Jaballah), conti@math.unipd.it (Mauro Conti), mosbah@labri.fr (Mohamed Mosbah), cpalazzi@math.unipd.it (Claudio E. Palazzi)

external communication infrastructure. However, to effectively broadcast an alert message from a vehicle involved in an accident to all the following vehicles in the car platoon, the transmission of the message should be done as quickly as possible.

Current approaches are generally based on intermediate neighboring nodes, since the transmission of the alert message through infrastructures on the road would add delays that could have fatal consequences on life loss, injuries, and vehicle damages. For this purpose, different alert messaging applications have been proposed to broadcast an alert message very fast [4, 5, 6, 7]. However, it is crucial for such data exchange to be resilient to security attacks in order not to lose its potential effectiveness in saving lives. Attackers might run malicious actions to inject false information or alarm, thus rendering ineffective the safety application [8, 9, 10, 11, 12]. Although many alert message applications have been thoroughly studied in the past years [4, 5, 6], most of the obtained results did not take the security threats into account. Instead, the benefits in terms of reduced number of vehicles involved in a chain accident, thanks to a properly working alert message broadcast, is clearly shown by [5].

Contribution. In this paper, we review the contribution of [13] and further thoroughly investigate the impact of the position cheating attack on a representative algorithm for alert messaging, i.e., the Fast Multi-hop Broadcast Algorithm (FMBA) [4] for vehicular safety application. First, we highlight the vulnerability of FMBA to the position cheating attack and we study the impact of this attack on this algorithm. We show that this weakness we found could be leveraged by an adversary in a very effective way. Then, we discuss a solution we developed for broadcasting safety related messages, which is both fast and secure against the position cheating attack. We named this solution Secure FMBA and we assessed its effectiveness with a thorough simulation study.

Organization. This paper is organized as follows. The next section presents the notation and model assumptions. Section III reviews the proposed solutions in the literature, and provides the necessary background information related to the FMBA protocol. In Section IV, we show the vulnerability of FMBA to the position cheating attack. In Section V, we present our position verification method. In Section VI, we evaluate the performances of FMBA under different position cheating attacks, and Secure FMBA with position cheating detection. Finally, in Section V conclusions are drawn.

2. Notation and Model Assumptions

In this section, we present the assumptions and the notation (Table 1) used in this paper. In our model, we assume that one malicious vehicle is on the network, in order to show the impact of the attack with a minimum number of attackers. We did not assume and model the presence of obstacles or buildings along the road. We considered a symmetric communication range: if a verifier vehicle V hears a verified vehicle P, then we can also assume that P can hear V.

Moreover, we assume that V does not know in advance its transmission range and that it communicates directly with the prover node P. Each vehicle knows its own location, for instance, using GPS that provides accurate informa-

tion about current time and actual position. All the vehicles belong to a Public Key Infrastructure [14, 15]; i.e., each vehicle has a public/private pair of keys and a unique identity certified by a Certification Authority. We assume that the certification authority corresponds to the government agency responsible to assign license plates: a vehicle can be used only if it is provided with a unique license plate, a PKI certificate associated to its plate ID, and the public key of the Certification Authority. We assume also that certificate revocation lists are updated at given time interval (e.g., daily) by the vehicle and stored in a local memory. The power and computational resources are supposed to be large enough for our application's requirements, and the network is loosely time synchronized.

Symbol	**Definition**
$CMBR$	Current Maximum Back Range
$CMFR$	Current Maximum Front Range
$LMBR$	Latest-Turn Maximum Back Range
$LMFR$	Latest-Turn Maximum Front Range
$MaxRange$	How far the transmission is expected to go backward before the signal becomes too weak to be intelligible
d	Distance between two vehicles
CW	Contention Window
$CWMax$	Maximum Contention Window
$CWMin$	Minimum Contention Window
TR	Transmission Range
$Hello$	Hello message
drm	Declared transmission range in the $Hello$ message
P	The prover vehicle
V	The verifier vehicle
R	The geographical region

Table 1: Notation

3. Related Work

The past decade has witnessed a growing interest in inter-vehicular communication (IVC) and its panoply of potential applications [16, 17]. IVC enables a vehicle to communicate with other vehicles, both directly and in a multi-hop fashion. Minimizing the broadcast delivery time is one of the main challenge for IVC. In the literature, it has been proven that this broadcast time is strictly related to both the number of relays of the messages (hops) and the network congestion [4, 6].

In order to minimize the number of hops that a message experiences during its propagation over the network, the approach in [18] assigns different contention windows to each vehicle receiving the message. The contention windows of the vehicles are inversely proportional to the distance from the previous sender. Each of these vehicles randomly selects a waiting time within its contention window before forwarding the message. These approaches generally

assume a unique, constant and well-known transmission range for all vehicles; unfortunately, this assumption is not realistic [4].

In [4], the authors propose a fast broadcast algorithm (FMBA). It aims at reducing the number of hops traversed by a message, in order to minimize its propagation delay. Vehicles in a car platoon dynamically estimate their transmission range and exploit this information to efficiently propagate a broadcast message with as few transmissions as possible. In essence, the farthest vehicle in the transmission range of a message sender or forwarder will be statistically privileged in becoming the next (and only) forwarder. In [6], authors have enhanced the fast broadcast algorithm using heterogeneous transmission range. Unlike [4], the authors select the forwarder of the message as the vehicle which transmission spans farther. In [6], the forwarder of a message is not chosen as the farthest vehicle in the transmission range of the sender.

The problem of providing security in vehicular communications has been a roadblock to their large scale deployment; yet it is still in its infancy. Accurate information on position is crucial for IVC based vehicular safety applications. In particular, several multi-hop broadcast algorithms have been proposed [4, 6, 19] without security in mind, whereas security is a fundamental problem in this context which should not be overlooked. Indeed, attackers might run malicious actions to inject false information or alarm, thus rendering ineffective the safety application [10]. To this aim, detection mechanisms have been proposed in this context to recognize nodes cheating about their location [20, 21]. Position verification approaches can be grouped into two main categories [20]: infrastructure based and infrastructure-less based approaches.

Infrastructure based approach. This class of approaches is based on special hardware dedicated infrastructure to verify the position of other vehicles. Some verification approaches are based on distance and angle estimation techniques. These techniques include time difference of arrival, time of arrival, received signal strength, and angle of arrival [22]. However, these methods are not secure in presence of attackers. In [23], the authors propose range-free protocols that do not require distance or angle measurements, but are also insecure in adversarial environments. The technique of distance bounding does not prevent an attacker from declaring a position farther away than it really is [24]. Moreover, in [25], they assume that each verifier is trusted, which is not realistic.

The solutions in [20] use multiple sensors to monitor and calculate trust values for position information. There are two classes of position verification sensors: autonomous and cooperating sensors. In fact, autonomous sensors work autonomously on each node and contribute their results to the overall trust ratings of neighbors. Cooperating sensors use the information exchange between the neighbors to verify positions. The solution in [26] uses verifiers at special locations. This solution needs specific infrastructure: the verifiers. These verifiers attempt to verify location claims for region R that are "near" a verifier V. In [21], the proposed solution depends on two directional antennas.

Each vehicle periodically sends a message containing its location together with its own two lists of front and back neighbors. A vehicle decides on the relative positions of its one-hop neighbors based on the messages it receives.

In [27], the authors refer to an infrastructure based scheme, which is a framework for tracking vehicles. This framework cooperates with nearby cars to collect location claims. An authority should verify the locations declared by vehicles. In [28], the authors devise a protocol using verification of location claims to detect sybil attacks in vehicular communications.

Infrastructure-less approach. Solutions of this type can be further classified in parameter based and model based approaches [20]. In parameter based approaches, vehicles check whether a claimed node's position is within a degree of accuracy from the real position. This check is based on acceptable values of some network and traffic parameters, such as i) packet's timestamps consistency with current time; ii) acceptance range which assumes that no neighbor is further than the maximum transmission range. This approach unrealistically assumes that the transmission range is fixed [29]. On the other side, model based solutions compare the regular behavior of the system and current actions to identify anomalies that could indicate malicious behaviors [30]. Each node periodically broadcasts its database containing information about observed nodes. When a broadcast is received, the content is merged into the receiving node's database. Each node periodically examines events in its database searching for the scenario with the least number of malicious nodes. The disadvantage of this class of solutions is that a big search space of possible scenarios is needed to ensure efficacy.

In this paper, we review the contribution of [13] and thoroughly investigate the impact on the position cheating attack to the Fast Multi-hop Broadcast Algorithm (FMBA) as a case study among the other used schemes [6, 7]. The analysis and the further set of simulations presented show that the position cheating attack degrades the performances of FMBA. Furthermore, we propose a detection mechanism to lessen this problem and to recognize nodes cheating about their position.

3.1. Fast Multi-hop Broadcast Algorithm: FMBA

The aim of Fast Multi-hop Broadcast Algorithm (FMBA) is to reduce the time required by a message to propagate from the source to the farthest vehicle in a certain area of interest [4]. To achieve this goal, FMBA exploits a distributed mechanism for the estimation of the communication range of vehicles. These communication range estimations are obtained by exchanging a number of *Hello* messages among the vehicles, and are then used to reduce the number of hops an alert message has to traverse to cover a certain area of interest. This leads to a decrease in the number of transmissions as well as the time required by a broadcast message to reach all the cars following the sender within a certain distance.

This scheme is composed by two phases: the estimation phase, and the broadcast phase. The former is continuously active and is meant to provide each vehicle with an up-to-date estimation of its transmission range. Instead, the latter one is performed only when a message has to be broadcast to all vehicles in the sender's area of interest. In order to forward a packet, each receiver has to compute its waiting time before attempting to forward the message. This waiting time is expressed through a contention window (CW) computed using Equation 1.

$$CW = \left| \frac{(MaxRange - d)}{MaxRange} \times (CWMax - CWMin) + CWMin \right|. \quad (1)$$

When a car has to send or forward a broadcast message, it computes the $MaxRange$ value in the broadcast message as the maximum between $LMBR$ and $CMBR$ values. To avoid unnecessary transmissions, all vehicles between the original sender and the current forwarder abort their attempt to forward the message; whereas all vehicles behind the current forwarder compute a new CW, based on last forward parameters, to participate in the election for the forwarder on the next hop.

4. Position-Cheating Attack

In this section, we show that FMBA is vulnerable to a position cheating attack. In particular, the goal of this attack is to induce a delay in the alert broadcast by increasing the Contention Window (CW) of honest vehicles.

To run the position cheating attack, a malicious node announces in a *Hello* message a false position, i.e., claiming to be farther away in the direction of the alert message (in this paper we assumed the direction to be backward). Hence, honest nodes receiving an alert broadcast message will compute unnecessarily large CWs, thus slowing down the forwarding process. For ease of presentation, Figure 1 depicts the impact of this attack reporting the CWs of some vehicles depending on their distance from the original sender/forwarder (vehicle V) of the alert message.

Since the CW of each vehicle is computed through Equation 1, without the malicious vehicle the CW function should vary as shown by the continuous line in Figure 1 (from its maximum in correspondence of vehicle V to its minimum at the end of the transmission range which is assumed to be close to vehicle D). Instead, if during the estimation phase, a malicious vehicle within V's transmission range sent a *Hello* message to declare a fake position corresponding to *M'* in Figure 1, the transmission range estimation of vehicle V would be wrongly computed as the distance from V to *M'*, instead of the distance from V to D. This leads vehicles A, B, C and D to wrongly compute their CWs with higher values. In fact, those nodes will consider the minimum CW in correspondence of position of *M'*, as shown by the dotted line in Figure 1.

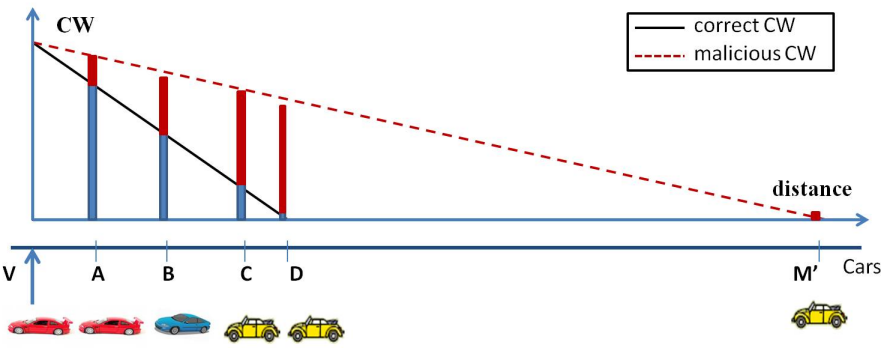


Figure 1: Impact of distance cheating on the waiting time

The described simple position cheating attack modifies the computation of CW, increasing in average the contention period of each node before any forwarding transmission can take place, hence slowing down the transmission of the alert message. Algorithm 1 describes more in details the attack as executed by a malicious vehicle M. In fact, M cheats about its claimed position, declaring a false one (Algorithm 1, line 2). Then, M broadcasts its *Hello* message (Algorithm 1, line 3) indicating its claimed position.

Algorithm 1: Position-Cheating attack executed by a malicious vehicle M

1 Input: *real_position*: (Real position of M);
claimed_position: (Claimed position of M);
vehicle_ID: *ID* of the vehicle M;
drm: declared max range of M;
Hello msg: *Hello* message generated by M;
2 $claimed\_position > real\_position$;
3 $M \rightarrow *$: Hello msg = $< vehicle\_ID, claimed\_position, drm >$;

5. Position Cheating Detection

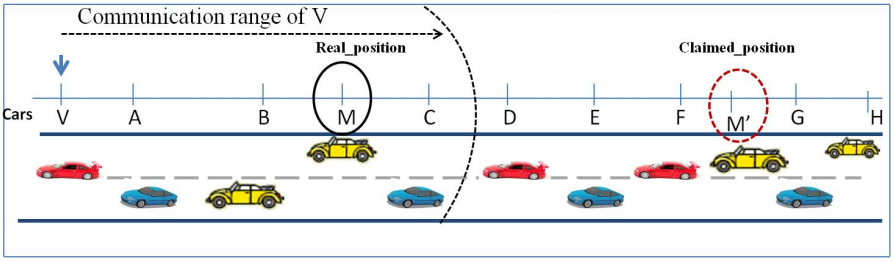
In FMBA protocol [4], information dissemination about vehicle positions is fundamental. Hence, vehicles (successfully) cheating about their position can have a severe impact regarding the performance and security of the algorithm. In this work, we propose a detection mechanism that is able of recognizing nodes cheating about their position. Unlike other proposals described in the literature [20], [26], our detection mechanism does not rely on additional hardware. Instead, our solution uses collaborative neighbors. We present an overview and a detailed description of our false position detection mechanism.

5.1. Overview

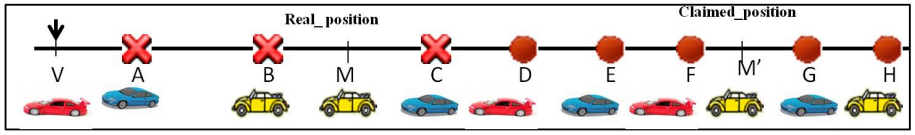
In this section, we present different position cheating attacks, differentiated by the claimed position and the real position of the verified vehicle. The entities involved are the verifier vehicle V, the prover vehicle M, and the other

vehicles in the road. M claims a position in the *Hello* message. The verifier vehicle V uses information, collected by collaborative vehicles, to decide whether M is a cheater or not. We distinguish three cases based on the real position and the claimed position of M.

In the first case (Case 1), as presented in Figure 2(a), M claimed a position M' that is not included in the actual transmission range of the verifier V. Vehicle V collects information from neighbor vehicles in order to decide whether the claimed position of M is fake or not. In Figure 2(c), we report the notation used to indicate whether a vehicle hears a message or not. This legend is also applied to the other following figures. In Figure 2(b), the reports of the vehicles demonstrate that only A, B and C have heard the node M. Based on this reported information, the verifier V can determine that M is cheating about its position.



(a) Case 1: Claimed position is not within the transmission range of the verifier



(b) Results of vehicles' reports

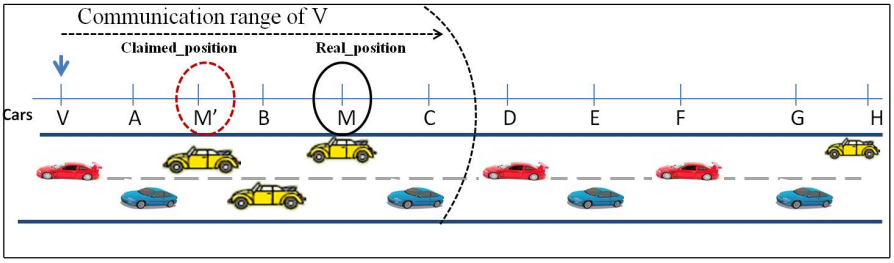
(c) Legend

Figure 2: Case 1

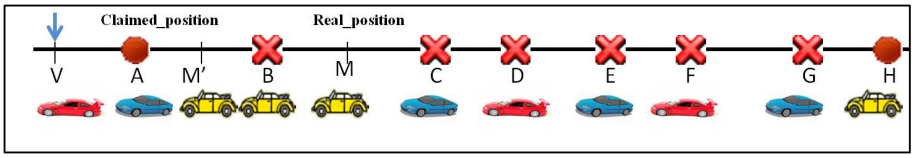
We discuss the second case (Case 2) using Figure 3(a). In this case, the claimed position of M (position M') is in the communication range of the verifier V. Vehicles A, B, and C report their information about their neighbors (Figure 3(b)). These reports confirm that the considered vehicles (A, B, and C) received M's message, whereas vehicles D, E, F, G, and H did not hear it. Based on the collected information, the verifier node decides that the received information is consistent.

In the third case (Case 3), depicted in Figure 4(a), the verifier V is located between the real position and the claimed position of M. The verifier, in the third case, could confirm that the reported information is consistent. Thus, M might be successfully cheating; yet, the impact of this false location does not lead to successfully modify the CW of V as the claimed position is between the positions of F and G, thus not affecting the computation of the maximum

transmission range (Figure 4(b)).

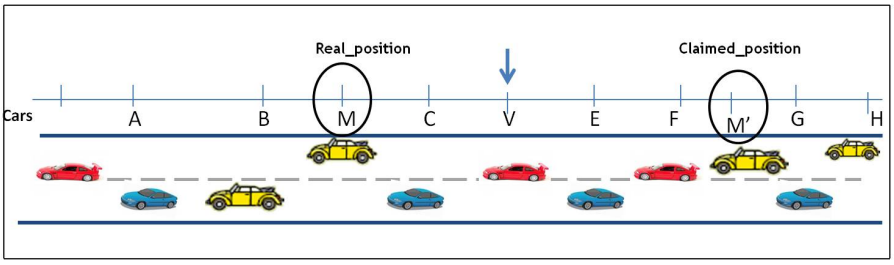


(a) Case 2: Claimed position and real position are within the transmission range of the verifier

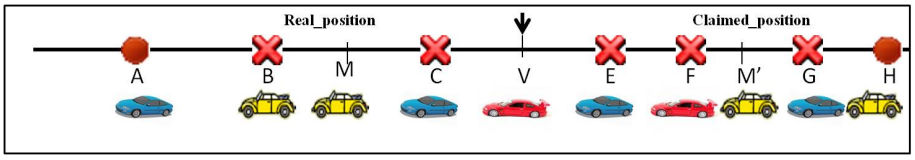


(b) Results of vehicles' reports

Figure 3: Case 2



(a) Real position and claimed position are on the transmission range of V



(b) Results of vehicles' reports

Figure 4: Case 3

We could summarize these cases in Figure 5, which represents V's CW as a function of the distance of the different neighbors. In fact, the CW is divided into three regions based on the different collected reports and position of vehicles. Let us consider the three regions denoted by ($R1$), ($R2$), and ($R3$). Region ($R1$) represents the regular CW values (represented by the continuous line) of the vehicle V without an attack. Furthermore, if M's claimed position exceeds C's position (for example the position M'_{R_2}), this information could have an effect on the waiting time of other vehicles (A, B, C); until the claimed position of the prover reaches the position of D (represented by the dotted line). If the claimed position of M exceeds D, in this situation the attack is detected. Region ($R2$) is limited by the

positions of C and D, then a claimed position of M in this Region ($R2$) has a small effect on V's CW. Region ($R3$) represents the position of vehicles superior than the position of D. Vehicles D, E, F, G, and H are in Region ($R3$). If a malicious vehicle claims to be in Region ($R3$) (for example M'_{R_3} in Figure 5), it will be detected by V because most of the vehicles in this region report the non overhearing of M. Thus, the verifier V will detect M as a cheater.

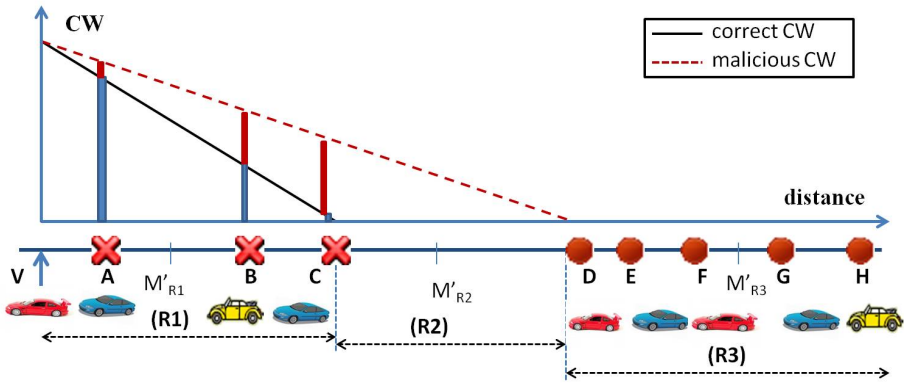


Figure 5: Verifier waiting time versus distance

5.2. *Description*

In this section, we describe our proposed position verification scheme. We discuss how the information of the vehicles could be propagated to other vehicles, in order to have a complete and a local observation. First, we discuss the structure of the transmitted message, and the timing of forwarding or transmitting the information. Second, these data could be collected from direct neighbors (in case nodes communicate directly) or from multi-hop neighbors (in case of indirect or asymmetric communication) between vehicles. Yet, we should limit the multi-hop propagation of this information within a limited region. Collected information should be recent and limited to the participating nodes. Third, in order to guarantee the authenticity of messages, nodes use an authentication mechanism. To do this, we propose to transmit the information of vehicles in a modified *Hello* message. After receiving the different reports (the modified *Hello* messages) from vehicles, each verifier executes locally a position verification procedure. Then, the verifier vehicle could detect whether the claimed position of a vehicle M is false or not.

5.2.1. *Structure of the modified Hello message*

In order to have the position of the vehicles and their neighbors, we propose to include these data into the *Hello* message instead of using additional structures. The first reason supporting this choice is that the *Hello* message is transmitted by a vehicle in each time interval (between 100 *ms* to 300 *ms*). A second reason is that the sender of the *Hello* message is chosen randomly. A third reason is to reduce the communication overhead and not generate another structure or another type of message. The modified *Hello* message includes the *vehicle_id*, the *vehicle_position*,

declared_max_range, *timestamp*, *list_neighbors* which is the list of two hop neighbors, and a signature generated by the sender private key. Figure 6 illustrates the structure of the modified *Hello* message. Each element (of type Neighbor) in the *list_neighbors* has *vehicle_id*, *vehicle_position*, and a list of indirect neighbors. In turn, a list of indirect neighbors is composed by a set of elements of type Indirect Neighbor, i.e., a structure which includes *vehicle_id*, *vehicle_position*, *declared_max_range*, and *timestamp*.

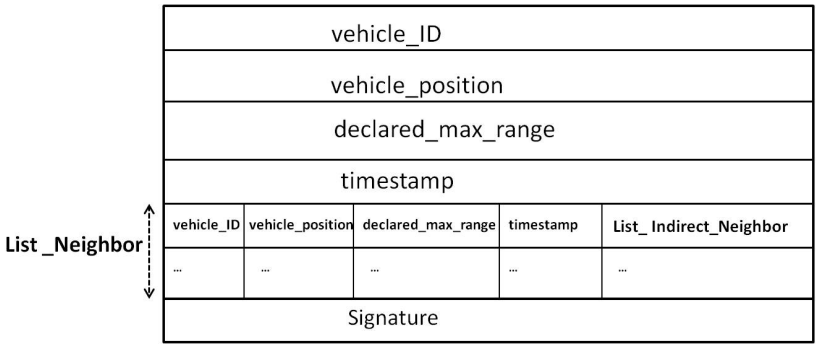


Figure 6: A modified structure of *Hello* message

5.2.2. Sending Hello message procedure

We focus on the *Hello* message sending procedure (Algorithm 2). In every turn, each vehicle determines a random waiting time (lines 1 and 2). After this waiting time, if neither other transmission is heard nor collision happened (line 6), it proceeds with transmitting a *Hello* message. This *Hello* message includes *vehicle_ID* (line 7), the *timestamp* (line 9), the *vehicle_position* (line 8), the *declared_max_range* (line 10), the list of neighbors and their two hop neighbors: *list_neigh_S* (line 11). Furthermore, the sender uses its private key to generate a signature for the message (line 12), then it transmits the message (line 13).

5.2.3. Receiving Hello message procedure

The *Hello* message receiving procedure is depicted in Algorithm 3. In particular, a vehicle receiving a *Hello* message in line 2, generates the public key (line 3) using $f(sender\_id\_X)$ where f is a hash function. In line 4 of Algorithm 3, the receiver verifies the signature of *Hello* message. Then, it checks for the freshness of message (line 6). Indeed, it could be an old message transmitted to the vehicles. This check is performed verifying the coherence between the time inserted in the message by the claiming vehicle (the sender of the *Hello* message) and the current time of the receiver. Then, for each message that passes the previous checks, the receiver vehicle extracts the information (*sender_id_X*). The receiver checks whether this is the first received *Hello* message carrying the *sender_id_X*. Then, the receiver simply stores (*sender_id_X*, *sender_position_X*, *declared_max_range*, *timestamp*, *list_neighbor_X*) to its list of neighbors (Algorithm 3, line 9).

Algorithm 2: Sending *Hello* message algorithm (executed by vehicle X)

```
1  Input: list_neighbors: list of neighbors, vehicle_X: the identity of the sender X, position_X: the sender position, current_Time_X: current
   time of the sender, LMFR, CMFR, list_neigh_X: the list of neighbors of X, K^X_private: private key of the sender S, H: hash function;
2  Output: Hello message ;
3  For each turn ;
4  sending_time := random(turn_size);
5  wait (sending_time);
6  if not (heard_Hello_msg() or heard_collision()) then
7      Hello_msg.vehicle_ID:= vehicle_X;
8      Hello_msg.vehicle_position:= position_X;
9      Hello_msg.timestamp:= current_Time_X;
10     Hello_msg.declared_max_range:= max(LMFR, CMFR);
11     Hello_msg.list_neighbor:= list_neigh_X;
12     Hello_msg.signature:= K^X_private
       (H(Hello_msg.vehicle_ID, Hello_msg.vehicle_position, Hello_msg.timestamp,Hello_msg.declared_max_range,
       Hello_msg.list_neighbor)) ;
13     transmit (Hello_msg);
14 EndFor
```

After that, the receiver determines its own position (line 12), extracts from the *Hello* message the *sender_position* (line 12), and the included estimation of the maximum transmission range (line 11), and determines the distance between itself and the sender (line 13). If the *Hello* message is received from ahead, the value of $CMFR$ is updated (lines 14 and 15), otherwise $CMBR$ is updated (lines 16 and 17). In both cases, the new value is obtained as the maximum among the old one, the distance between the considered vehicle and the *Hello* message sender, and the sender's transmission range estimation provided by the *Hello* message.

Algorithm 3: Receiving *Hello* message algorithm (executed by vehicle V)

```
1  Input: list_neighbors: list of neighbors of V,current_Time_V: the current time of V, sender_id_X: the identity of the sender,
   sender_position_X: the field corresponding to sender position, currentTime_X: current time of the sender included in the message, drm_X:
   the declared maximum range received, list_neigh_X: the list of neighbors in the received message, signedHelloMsg_X: the received
   signature ;
2  < sender_id_X, sender_position_X, currentTime_X, drm_X,
   list_neigh_X, signedHelloMsg_X > ;
3  K^Pub_X ← f(sender_id_X);
4  if H(sender_id_X, sender_position_X, currentTime_X,
   drm_X, list_neigh_X) ≠ K^Pub_X(signedHelloMsg_X) then
5      handle this exception ;
6  if IsNotCoherent (current_time_X, current_time_V) then
7      handle this exception ;
8  if IsNotPresent (list_neighbors, sender_id_X) then
9      add(list_neighbors, < sender_id_X, sender_position_X,
       currentTime_X, drm_X, list_neigh_X, signedHelloMsg_X >)
10 mp := my_position() ;
11 sp := sender_position ;
12 drm := declared_max_range ;
13 d := distance(mp, sp) ;
14 if (received_from_front(Hello_msg)) then
15     CMFR := max(CMFR, d, drm) ;
16 else
17     CMBR := max(CMBR, d, drm) ;
```

5.2.4. Position Verification Procedure

After receiving reports (*Hello* messages) from other vehicles, a vehicle V could decide whether the claimed position announced by a vehicle is correct or not. In Algorithm 4, we present the position verification algorithm executed by a vehicle V. In fact, V collects N reports (Algorithm 4, line 3). For each received report, it checks whether the claimed vehicle M is in the list of neighbors of these vehicles. Based on this information, V could decide if the claimed position is in a certain Region. If the claimed position is in Region ($R1$) (line 9), then the vehicle could be a malicious one, but this has no negative effect on V. Otherwise, if the claimed position is in Region ($R2$) (line 12), then the position of M has an effect on V, and M is classified as suspicious. Finally, if the claimed position is in Region ($R3$) (line 15), M is detected as a cheater. If there is at most one malicious node, we refer to the Case 1.

Algorithm 4: Position verification algorithm (executed by vehicle V)

```
1  Input: V: the verifier node executes the verification algorithm
   M: the vehicle for which V wants to verify its claimed position
   M_1,...M_N: N messages collected by V; Claimed_position of M
   M_i =< sender_i, sender_position_i, timestamp_X,
   drm_X, list_neigh_i >;
2  Output: State of M is malicious or suspicious;
3  //i is the sender of the report M_i
   For all i ∈ 1, ...Ndo
   extract (list_neigh_i) from the report M_i;
4  if M ∈ list_neigh_i then
5  |   i hears M ;
6  else
7  |   i does not hear M ;
8  End For.
   V sets its CW with respect to the different information;
9  if claimed_position ∈ Region (R1) of V then
10 |   M is not detected as malicious and the claiming distance has no effect on V;
11 else
12 |   if claimed_position ∈ Region (R2) of V then
13 |   |   M is not detected and has an effect on V ;
14 |   else
15 |   |   if claimed_position ∈ Region (R3) of V then
16 |   |   |   M is detected as malicious;
```

6. Performance Evaluation

We carried out an exhaustive experimental study to test FMBA under the position cheating attack, and compare it with Secure FMBA using a position cheating detection mechanism. We use for our experiments the well known NS-2 simulator (version ns-2.29) [31]. We use the wireless model two-ray ground reflection on a highway with multiple lanes. Configuration parameters are summarized in Table 2.

In our simulations, we focus on answering to the following nine questions: 1) What is the distance declared by the malicious vehicle? 2) Does the attack delay the transmission of the alert message? 3) What is the correlation between

Parameter	Value
Hello Message Size	50 B
Broadcast Message	200 B
Idle Time	100 ms
Time Slot	200 μs
$CWMin$	32
$CWMax$	1024
Vehicle Speed	70 – 140 km/h
Simulation Time	40 s

Table 2: Configuration Parameters

the average number of slots and different transmission ranges? 4) Does our Secure FMBA reduce the average number of slots waited for, before forwarding an alert message? 5) Is there a correlation between the number of retransmissions and the average number of slots? 6) How many hops are needed to spread an alert message over the area of interest? 7) Does the attack have an impact on the estimated transmission range of vehicles? 8) Are all the vehicles affected by the attack? 9) Does Secure FMBA correctly estimate the transmission range of the vehicles?

For the sake of clarity, this attack could be led by a terrorist in order to kill or injure a person. Moreover, insurance companies might loose a lot of money in case of accidents. Thus, they might be interested in introducing the system on the "black box" of the cars, in order to save money. Similar systems are already in place to reduce the insurance cost while tracing the position and speed of cars. If accident alerting systems such as FMBA can be proven to effectively work (despite malicious behaviours), this might allow a new business model for car insurances, and also reducing cost of insurance for the customers.

Finally, we note that different malicious vehicles could execute also a collaborative attack. Indeed, we could distinguish two major scenarios. One scenario consists on having all the malicious vehicles on the same transmission range. The impact of the attack will be the same as having one malicious vehicle on the platoon. Another interesting scenario considers that malicious vehicles are under different transmission ranges, i.e., under different hops. Thus, the impact of the attack will be multiplied by the number of hops that are affected by the malicious vehicles.Given the above discussion, in the remaining part of this section we focus only on the attack run by a single malicious vehicle, while we leave as a future work a thorough experimental evaluation of the impact of a bigger number of adversarial vehicles.

6.1. Simulation Environment

To provide a general analysis of the implemented protocols, not affected by the implementation details (e.g., 802.11b/g/p or others), we use a discrete progression of the time based on slots to represent all the time related variables (such as MAC layer contention windows, random waiting, and time intervals). In particular, we measure the time variables in slots as done in [4]. We implement and evaluate the performances of different protocols: the

original FMBA; FMBA with four different attacks; and Secure FMBA. We carried out an extensive set of scenarios, and for each of them we run 100 simulations averaging their outcomes to produce charts. For each scenario, we use three different transmission ranges TR =300 m, TR = 650 m, and TR = 1000 m. We choose 300 m and 1000 m as transmission range since these values come straightforward from the $IEEE$ 802.11p draft, which indicates these two values as the boundaries for a highway scenario [32]. We also choose the value of TR = 650 m as an intermediate value between TR = 300 m and TR = 1000 m to show how our protocols scale.

Before the forwarding process, every vehicle in the platoon computes a random waiting time within the contention window. The contention window is initially set to $CWMin$, and follows a general backoff mechanism. Using the back-off mechanism, every vehicle doubles its value every time a transmission attempt results in a collision and decreases linearly with every successful transmission. The first vehicle in the platoon generates an alert message and forwards it after 37 seconds of simulation. In particular, a *Hello* message is transmitted every 100 ms to feed the transmission range of the vehicles. In our simulations, we consider a malicious node sending only one false *Hello* message and then cheating about its position, thus impacting on the transmission range estimation of some vehicles. An alert message is then transmitted. This allows us to evaluate the impact of the false position information (declared by the malicious vehicle).

FMBA uses a low transmission rate since the protocol generates an alert message only when an abnormal behavior happens. Thus, the overhead due to *Hello* messages is very limited, i.e., less than 1 Kb/s within a transmission range area. The alert message has to be propagated from a certain vehicle to all following vehicles in an area of interest of $7 \times TR$ (TR is the Transmission Range). The number of vehicles per Km of (multi-lane) road varies from 15 to 400, representing different density levels. In our evaluation, we consider ten different scenarios: i) FMBA without attacks; ii) FMBA with a false position cheating of 1.5 TR; iii) FMBA with a false position cheating of 3 TR; iv) FMBA with a false random position cheating between 0 and 6 TR; v) FMBA with a false position cheating of 5 TR; vi) Secure FMBA without attacks; vii) Secure FMBA with a false position cheating of 1.5 TR; viii) Secure FMBA with a false position cheating of 3 TR; ix) Secure FMBA with a false random position cheating between 0 and 6 TR; and x) Secure FMBA with a false position cheating of 5 TR.

6.2. What is the Distance Declared by the Malicious Vehicle?

In our simulations, the malicious node cheats about its position, by claiming either a fixed position or a random position. We considered a scenario where the attacker declares a random position between 0 and 6 TR. In Table 3, we report the average and the standard deviation (STD) of declared distance by the attacker when running the random attack, where Dens. refers to vehicles per km. It is worth noting that the average of declared distance does not depend on the vehicle density.

Dens.	**TR** = 300 m		**TR** = 650 m		**TR** = 1000 m	
	Average	STD	Average	STD	Average	STD
15	899.3	570.6	1883.5	1121.1	3110.1	1742.1
25	827.8	472.9	1895.6	1254.1	3048.1	1765.9
50	1014.1	572.3	1984.3	1179.9	3316.1	1732.8
100	935.6	537.9	2070.7	1110.0	2936.6	1670.5
200	830.3	535.9	2150.1	1034.3	2957.1	1857.8
300	870.7	562.6	2009.3	1195.5	3561.1	1799.4
400	908.9	500.7	1881.2	1041.3	3228.3	1765.4

Table 3: Declared Random Attacked Distance

6.3. Does the Attack Delay the Transmission of the Alert Message?

An important question entails deriving for how long, on average, an alert message waited before its transmission to the end of the platoon. In Figure 7, we report the average number of slots required to broadcast a message. In the x-axis, we present the vehicle density, and in the y-axis, we present the number of slots. We evaluate the performances of FMBA (without attack), and FMBA with the four different cheating positions (d = 1.5 TR, d = 3 TR, d = rand(0, 6 TR), and d = 5 TR). In Figure 7, for FMBA, with d = 1.5 TR (FMBA, d = 3 TR), we see that the malicious node cheats about its position and declares a fake one which is d = 1.5 TR (d = 3 TR) respectively, when running FMBA. For FMBA, with d = rand(0, 6 TR) we have a malicious node which declares a random fake position between 0 and 6 TR. A FMBA, with d = 5 TR represents a cheating position of 5 TR.

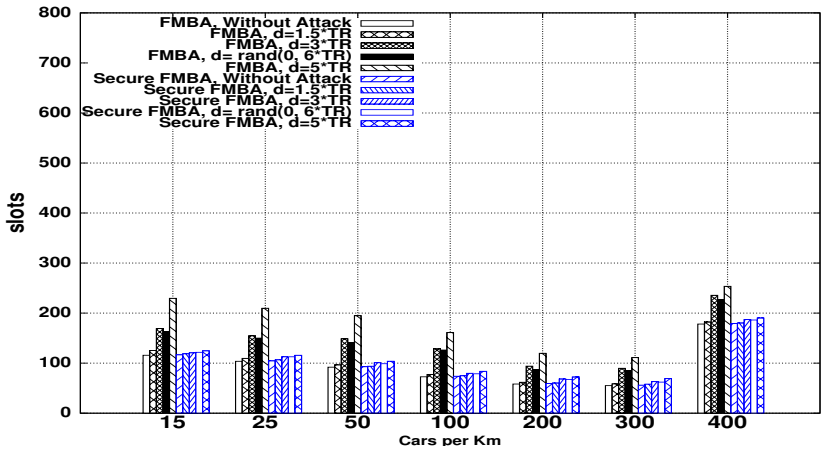
Figure 7(a) represents the average number of slots for the different protocols when using a transmission range TR= 300 m. Let us focus on FMBA algorithm. From this figure, we observe three interesting outcomes. First, with a very low vehicle density we have a reduced precision of the transmission range estimation (this would very slightly increase the number of slots). As a confirmation, when vehicle density is 15 cars per km, the number of slots is 115, and this number decreases slightly when the density is 50 and 100 vehicles per km. Moreover, from Figure 11, we can confirm this decrease since the estimation of the transmission range for all the protocols has a lower precision when vehicle density is low. Let us keep this information in mind: We justify this in our answer to Question 8 in Section 6.9.

Second, with higher vehicle densities, we first have a decrease in the total number of slots due to the fact that with more vehicles at the end of the platoon there will be a higher probability for one of them to choose a small random value within their contention windows. However, with very high vehicle densities, the number of slots needed in average to reach the end of the platoon has a big increase. This is due to the high chance for two vehicles to transmit

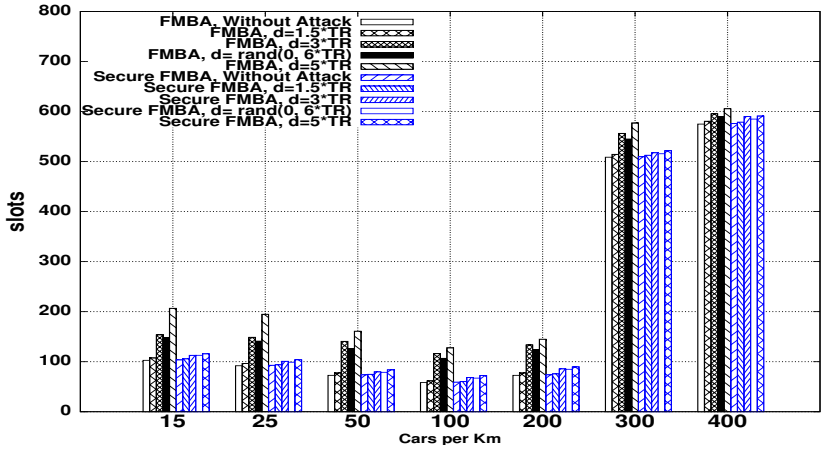
simultaneously thus leading to time-wasting message collision and retransmission. For instance, with a density of 400 cars per *km*, the number of slots in average is approximately 178 when $TR = 300\ m$. Similar trends are present in the outcomes of the different protocols in Figure 7(b), and Figure 7(c).

Let us focus on the impact of the four attacks on FMBA when the malicious node cheats about its position ($d = 1.5\ TR$, $d = 3\ TR$, $d = rand(0, 6\ TR)$, and $d = 5\ TR$). First, when an attacker runs a position cheating of $1.5\ TR$, we see that the degradation of the performances of FMBA is negligible (no more than few slots). For example, in Figure 7(a), when the density is 100 vehicles per *km*, the FMBA protocol with a position cheating of d=$1.5\ TR$ has approximately the same number of slots as FMBA without attack. Thus, the attacker that claims d=$1.5\ TR$, has a negligible impact on increasing the average number of slots.

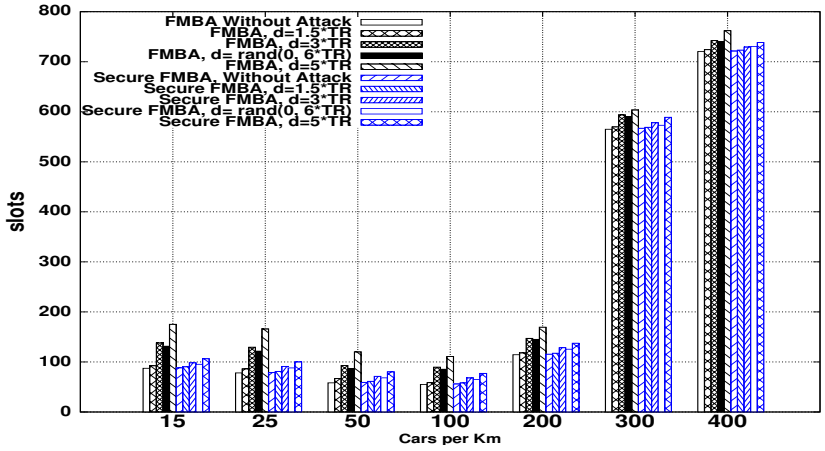
It is clear that the performance of FMBA degrades when the attacker declares a false position very distant from its real position. For example, FMBA with a position cheating of $d = 5\ TR$ has the worst impact among the evaluated attack scenarios. In this case, the number of slots increases more as compared to an attack with $d = 1.5\ TR$. As a consequence, the transmission of the alert message gets delayed more, as compared to an attack with $d = 1.5\ TR$. For instance, in Figure 7(c), when the vehicle density is 100, the malicious node with a cheating position of $d = 5\ TR$ adds more than 50 slots, compared to FMBA without attack. Another interesting point, is that the performances of the attacker are almost the same when the attacker runs a fixed false position of $d = 3\ TR$, and $d = rand(0, 6\ TR)$. The explanation for this is that even the attack is in average the same, however the impact of choosing values less than $3\ TR$ decreases the total number of slots.



(a) $TR = 300\ m$



(b) $TR = 650\ m$



(c) $TR = 1000\ m$

Figure 7: FMBA and Secure FMBA under attacks: Average Number of Slots

6.4. What is the Correlation Between the Average Number of Slots and Different Transmission Ranges?

An interesting point lies in understanding the correlation between the average number of slots for different transmission ranges. From Figure 7, we notice that the number of slots required to transmit a message with a transmission range $TR = 1000\ m$ is less than the protocols using a transmission range of $TR = 300\ m$ or $TR = 650\ m$. For instance, Figure 7(b) needs less slots to transmit the alert message since it uses a transmission range of $TR = 650\ m$, compared to the plotted protocols in Figure 7(a) using a $TR = 300\ m$. For example, when density is 15 vehicles per km, FMBA under $TR = 650\ m$ needs in average 102 slots. However, when using FMBA under $TR = 300\ m$, the message needs more than 115 slots to be propagated. From these figures, we can confirm that a high transmission range helps in quickly transmitting an alert message.

6.5. Does Our Secure FMBA Reduces the Average Number of Slots Waited for an Alert Message before its Forwarding?

We investigate on evaluating the average number of slots waited using Secure FMBA. Let us focus on Figure 7. For all the different transmission ranges ($TR = 300\ m$, $TR = 650\ m$, and $TR = 1000\ m$), Secure FMBA achieves lower transmission delays compared to FMBA under position cheating attack. This can be explained by the fact that Secure FMBA uses complete view based on local observations of vehicles, and each node is a verifier of a claim. When receiving a *Hello* message from a vehicle, the verifier vehicle will perform the verification procedure. Let us focus on Figure 7(c). We see that Secure FMBA for a vehicle density of 100 cars per km, and with a position cheating attack of $d = 5\ TR$, requires 30 slots less than the FMBA with $d = 5\ TR$. Moreover, we observe that the performances of Secure FMBA under different cheating positions are slightly the same. This could be motivated by the fact that Secure FMBA takes into account the position of vehicles, and whether they hear the message of the prover. We can confirm that the update of the transmission range of a vehicle is required when the prover is honest. However, as the suspicious region is very limited, thus it modifies slightly the estimated transmission range of the verifier.

6.6. Is There a Correlation Between the Number of Retransmissions and the Average Number of Slots?

Let us focus on the impact of the percentage of retransmissions/collisions on the average number of slots. Figure 8 shows the percentage of collisions in function of the vehicle density. We see that the percentage of collisions is low for low density of vehicles, and increases slightly when the vehicle density increases. In fact, when the vehicle density increases too much, then the possibility for two or more vehicles to randomly choose the same smallest number of slots increases as well, thus leading to time-wasting collisions and retransmissions.

Another interesting outcome from Figure 8 is that fixing a vehicle density, the number of retransmissions that occurred when the transmission range is $1000\ m$ is higher, compared to the other scenarios, i.e., when using a $TR = 650\ m$

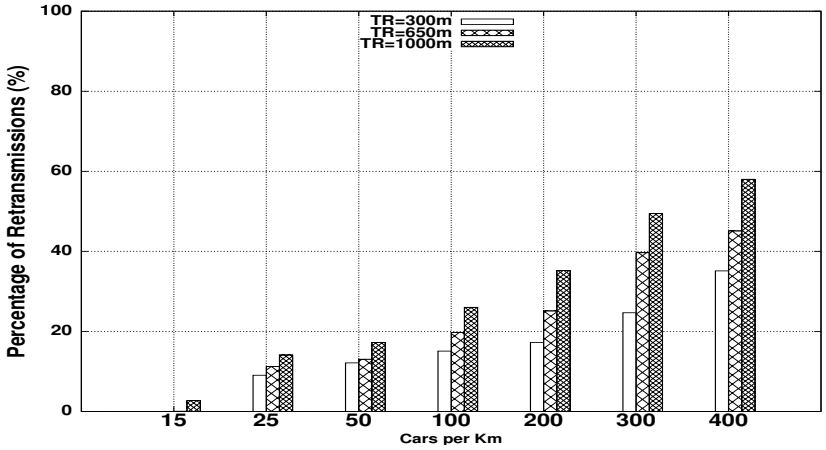


Figure 8: Retransmissions

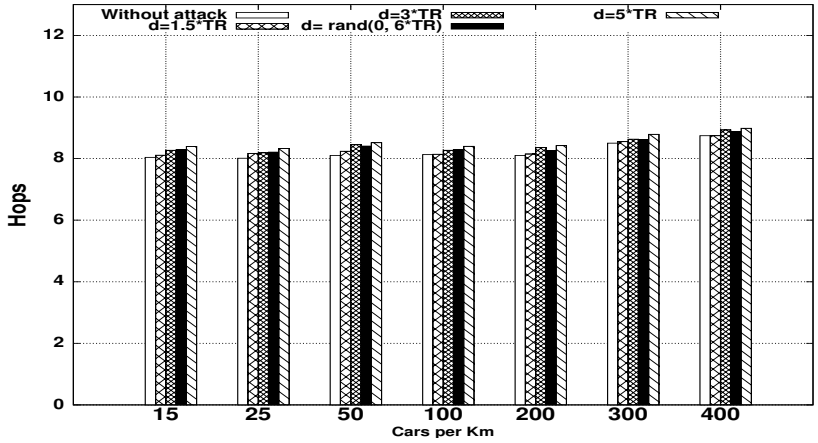
or TR=300 m. We see that when we increase the transmission range, and we have the same density of vehicles, there is a higher possibility that more than two nodes randomly choose the same smallest number of slots, thus generating collisions. For TR = 1000 m, and with a density of 200 vehicles per kilometer, the percentage of retransmissions is superior than 30%. However, for the same density, and having TR = 300 m, the percentage of collisions is less than 20%. The results in Figure 8 confirm our findings on increasing the number of slots in Figure 7, for a high vehicle density. In fact, the increase of slots when vehicle density is 200 cars per km, 300 cars per km, and 400 cars per km, for all the scenarios with TR smaller than 1000 m and 650 m is due to the increase of the number of retransmissions (see Figure 7).

6.7. How Many Hops are needed to Spread an Alert Message over the Area of Interest?

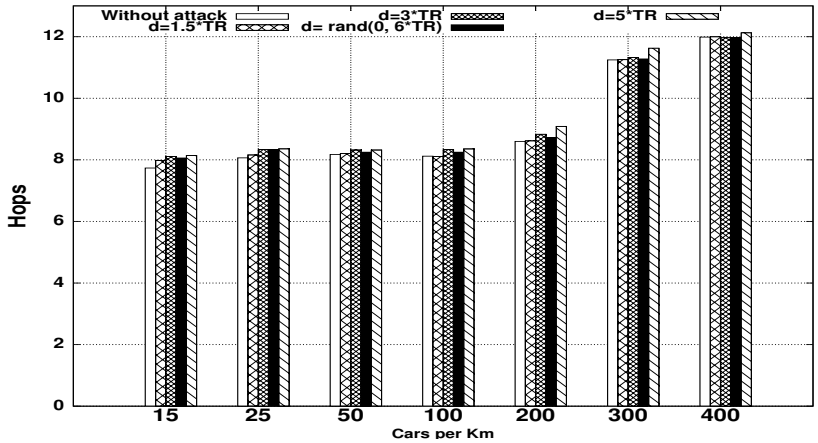
We study the average number of hops that a broadcast message experiences before covering the whole considered portion of road. In Figure 9, we report the number of hops. In particular, let us consider the scenario where the actual transmission range is 300 m (see Figure 9(a)). The x-axis represents the vehicle density and the y-axis shows the number of hops. It is interesting to note that the impact of the cheating position in this scenario does not affect the number of hops. The explanation of this is that we considered only one malicious vehicle, thus the attack affects approximately one hop. It is worth noting that, for TR = 300 m, the number of hops for each scheme varies slightly when augmenting the vehicle density (for densities higher than 400 cars per km).

Let us now focus on Figure 9(b). In this case, the number of hops increases when the density is 300 cars per km and higher. This happens thanks to the fact that the election of the next forwarder depends on vehicle's positions within the sender's transmission range and also chosen randomly. We should remind that the election of the forwarder on the broadcast phase of FMBA [4] depends on the value of the computed waiting time of each receiver (using Equation 1).

For a high vehicle density, the forwarder is not all the time at the end of the transmission range, thus the number of hops increases.



(a) $TR = 300\ m$



(b) $TR = 1000\ m$

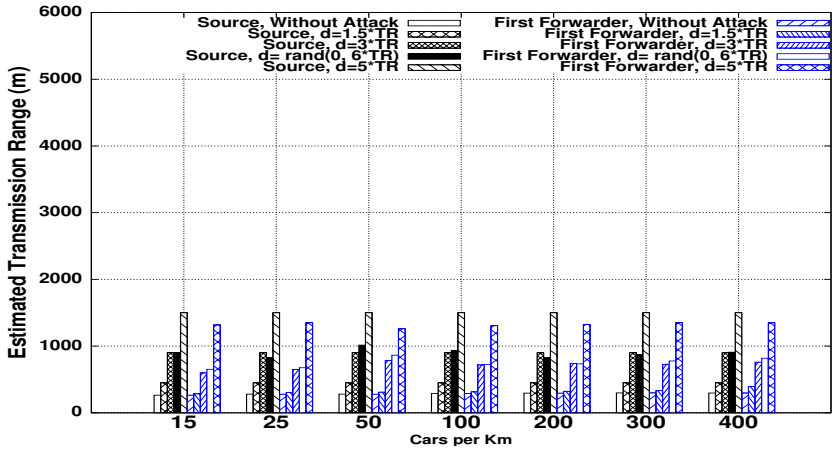
Figure 9: FMBA under the Position Cheating Attack: Average Number of Hops

6.8. Does the Attack Have an Impact on the Estimated Transmission Range of Vehicles?

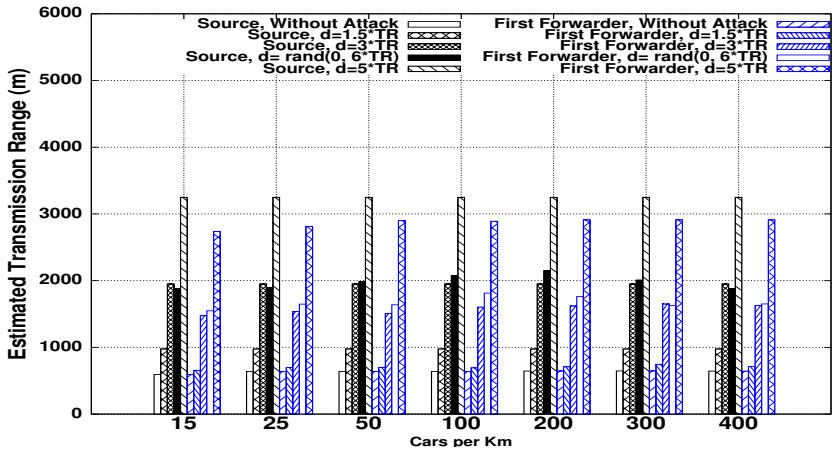
We study the impact of the false position cheating on the vehicles' estimation of their transmission range. Using a wrong transmission range parameter has consequences on propagation delays. In Figure 10, we report the estimated transmission range for both the sender and the first forwarder of the message. All the vehicles that are at one hop from the source, including the first forwarder, are affected by this attack, which increases their estimated transmission range. In the x-axis, we represent the vehicle density, whereas the y-axis represents the estimated transmission range. In particular, after a position cheating attack, the estimated transmission range is larger than with no attack. For the

first forwarder, the cheating position attack with $d = 1.5\ TR$ is not as effective as with greater values of d and generates almost the same transmission range estimation as using FMBA without attacks. In contrast, the estimated transmission range increases significantly with a cheating position set as $5\ TR$. For a vehicle density of 200 vehicles per km, and with a position cheating of $d = 5\ TR$, the average estimated transmission range is around 2900 m. Furthermore, the two protocols under attack (FMBA, $d = 3\ TR$ and FMBA, $d = rand(0, 6\ TR)$) have approximately the same transmission range estimation.

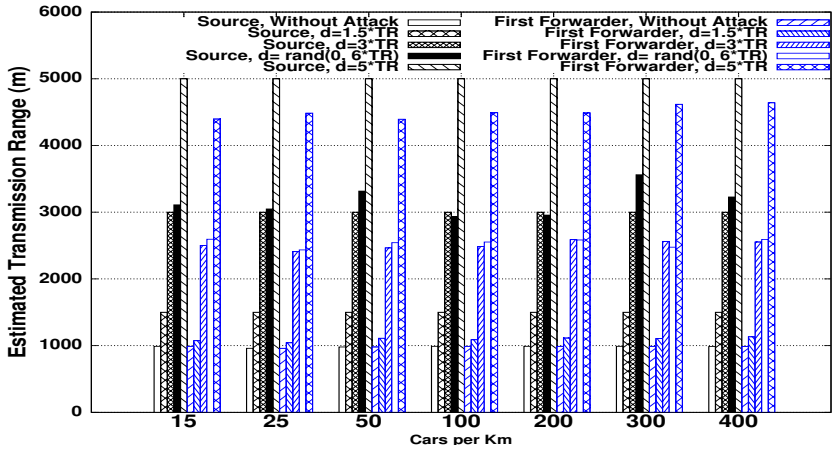
An interesting point in Figure 10 is the position of the forwarder of the message that was affected by the attack, for protocols within transmission range of $TR = 300\ m$, $TR = 650\ m$, and $TR = 1000\ m$, respectively. For more details, we refer the reader to Figure 10(a), Figure 10(b), and Figure 10(c). The property that emerges from these charts is that the first forwarder is located approximately on the middle of the transmission range of the sender. This is because the estimation is not made by the first vehicle (the sender) but by the first forwarder. This forwarder should be at the end of the transmission range. In case of attack, all vehicles will use high contention windows, thus providing all receiving nodes more or less the same probability to become the next forwarder. This means that the next forwarder will be in average at the middle of the transmission range, which is the difference between the values in the sender and the first forwarder in the charts (see Figure 10(a), Figure 10(b), and Figure 10(c)).



(a) $TR = 300\ m$



(b) $TR = 650\ m$



(c) $TR = 1000\ m$

Figure 10: FMBA under Position Cheating Attack

6.9. Are All the Vehicles Affected by the Attack?

In Figure 11, we report the estimated transmission range for the vehicles that are not affected by the attack. We note that the non attacked forwarders have the same value of the estimated transmission range as without attack. We see that for a low density of vehicles, there is not a precise estimation of the transmission range. These imprecise estimated transmission ranges imply that the number of slots will decrease for low vehicle density, until having a sufficient number of vehicles, that allows a good estimation of the transmission range. This explains the decrease of the number of slots in Figure 7 for all the different protocols, when the density of vehicles is low. An another interesting feature that emerges from Figure 7, is that FMBA as well as FMBA with position cheating attack use approximately the same estimation of transmission range (for the non affected vehicles by the attack). These vehicles will not be affected by the wrong estimation of the transmission range executed by the malicious node.

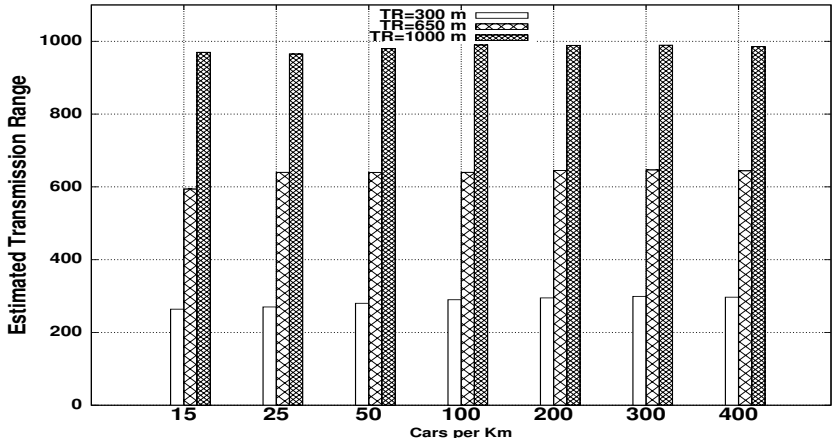
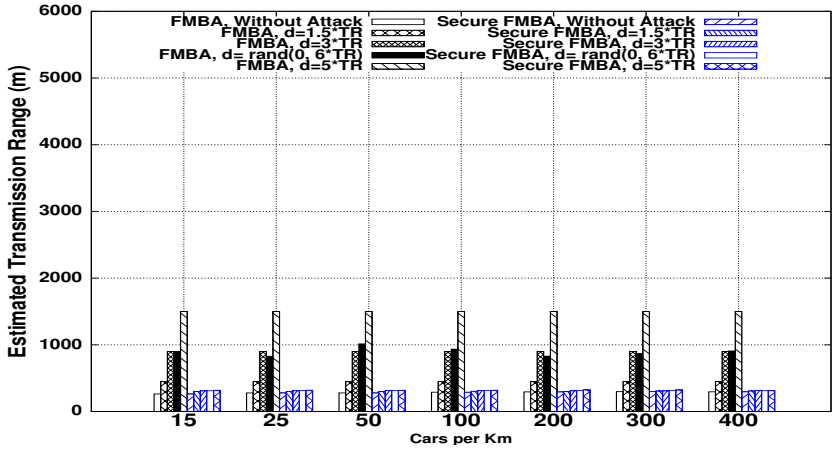


Figure 11: Estimation of the transmission range for non attacked vehicles

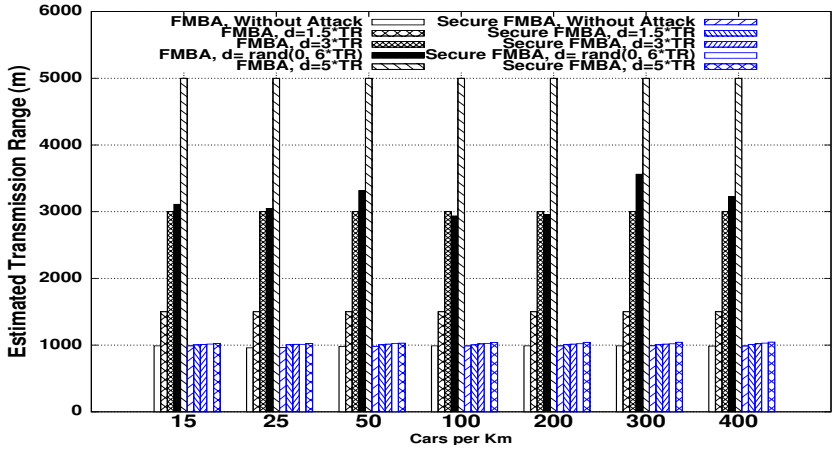
6.10. Does Secure FMBA Correctly Estimate the Transmission Range of the Vehicles?

We investigate on evaluating the estimated transmission range of the vehicles using Secure FMBA. In Figure 12, we report the estimated transmission range of the source under FMBA with attacks, and under Secure FMBA. The estimated transmission range of the source under FMBA with attacks depends on the position claimed by the malicious vehicle. In fact, in FMBA with attacks, the vehicle updates its transmission range as the maximum among: i) the distance from the sender to the receiver; ii) the broadcast $CMFR$ value; and iii) the previous value of $CMBR$. As the claimed position of the attacker is very high compared to its real position, then the receiver will update its transmission range based on the cheating position declared by the malicious vehicle. This explains the increase of the estimated transmission range of the different protocols under attacks.

Let us focus on Secure FMBA. For all the different cheating positions, vehicles under Secure FMBA have approximately a good estimation of the transmission range. For instance, when the factual transmission range is 300 m, the estimated transmission range of the source using Secure FMBA is less than 400 m. An explanation to this is that Secure FMBA uses the position detection mechanism, and does not update the transmission range of a vehicle when receiving a claim from a malicious one. This explains the decrease of the total number of slots in Secure FMBA compared to the attacked scenarios (see Figure 7).



(a) $TR = 300\ m$



(b) $TR = 1000\ m$

Figure 12: Estimated Transmission Range for the Source

7. Conclusion

In this paper, we have filled an important gap in safe and secure vehicular communications. We have highlighted the impact of a position cheating attack on FMBA, a state of the art fast broadcast algorithm. Furthermore, we have proposed a countermeasure for this threat, developing a solution which is both fast and secure against position cheating in broadcasting safety related messages. We have confirmed via experiments that FMBA is vulnerable to the cheating position attack, and carried out an extensive experimental study to assess the effectiveness of our solution.

8. Acknowledgments

This study has been carried out with financial support from the French State, managed by the French National Research Agency (ANR) in the frame of the Investments for the future Programme IdEx Bordeaux (ANR-10-IDEX-03-02). Mauro Conti is supported by a Marie Curie Fellowship funded by the European Commission for the PRISM-CODE project (Privacy and Security for Mobile Cooperative Devices) under the agreement n. PCIG11-GA-2012-321980. This work has been partially supported by the TENACE PRIN Project (n. 20103P34XC) funded by the Italian Ministry of Education, University and Research. This work is also partially supported by the MIUR/PRIN ALTER-NET and the UNIPD/PRAT Web Squared projects.

References

[1] http://www.who.int/en/.

[2] http://ec.europa.eu/transport/roadsafety/specialist/statistics/.

[3] C. Wu, Y. Liu, Queuing network Modeling of Driver Workload and Performance, IEEE TITS 8 (3) (2007) 528–537.

[4] C. E. Palazzi, S. Ferretti, M. Roccetti, G. Pau, M. Gerla, How Do You Quickly Choreograph Inter-Vehicular Communications? A Fast Vehicle-to-Vehicle Multi-Hop Broadcast Algorithm, Explained, IEEE CCNC, 2007.

[5] C. E. Palazzi, M. Roccetti, S. Ferretti, An Intervehicular Communication Architecture for Safety and Entertainment, IEEE TITS 11 (2010) 90–99.

[6] A. Amoroso, M. Ciaschini, M. Roccetti, The farther relay and oracle for VANET. preliminary results, IEEE WICON, 2008.

[7] M. D. Felice, A. Ghandour, H. Hartail, L. Bononi, Enhancing the performance of safety applications in IEEE 802.11p/WAVE Vehicular Networks, IEEE WOWMOM, 2012.

[8] L. Zhang, Q. Wu, A. Solanas, J. Domingo-Ferrer, A Scalable Robust Authentication Protocol for Secure Vehicular Communications, IEEE TVT 59 (2010) 1606–1617.

[9] B. Mishra, P. Nayak, S. Behera, D. Jena, Security in vehicular adhoc networks: a survey, IEEE ICCCS, 2011.

[10] G. Calandriello, P. Papadimitratos, J. P. Hubaux, A. Lioy, On the Performance of Secure Vehicular Communication Systems, IEEE TDSC 8 (2011) 898–912.

[11] Q. Wu, J. Domingo-Ferrer, Ú. González-Nicolás, Balanced Trustworthiness, Safety and Privacy in Vehicle-to-Vehicle Communications, IEEE TVT 59 (2) (2010) 559–573.

[12] Z. Ma, F. Kargl, M. Weber, Measuring long-term location privacy in vehicular communication systems, Elsevier Computer Communications 33 (2010) 1414–1427.

[13] W. B. Jaballah, M. Conti, M. Mosbah, C. E. Palazzi, Secure verification of location claims on a vehicular safety application, ICCCN, 2013.

[14] W. Diffie, M. E. Hellman, New directions in cryptography, IEEE TIT 22 (6) (1976) 644–654.

[15] R. L. Rivest, A. Shamir, L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, In ACM Communications Magazine 21 (2) (1978) 120–126.

[16] G. Karagiannis, O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin, T. Weil, Vehicular Networking: A Survey and Tutorial on Requirements, Architectures, Challenges, Standards and Solutions, IEEE Communications Surveys & Tutorials 13 (4) (2011) 584–616.

[17] H. Lu, C. Poellabauer, Balancing broadcast reliability and transmission range in vanets, IEEE VNC, 2010.

[18] E. Fasolo, R. Furiato, A. Zanella, Smart broadcast algorithm for inter vehicular communication, IEEE WPMC, 2005.

[19] H. Yoo, D. Kim, Repetition-based cooperative broadcasting for vehicular ad-hoc networks, Computer Communications 34 (15).

[20] T. Leinmüller, C. Maihöfer, E. Schoch, F. Kargl, Improved security in geographic ad hoc routing through autonomous position verification, ACM VANET Workshop, 2006.

[21] Z. Ren, W. Li, Q. Yang, Location Verification for VANETs Routing, IEEE WIMOB, 2009.

[22] D. Niculescu, B. Nath, Ad hoc positioning system (APS) using AoA, IEEE INFOCOM, 2003.

[23] D. Niculescu, B. Nath, DV based positioning in ad hoc networks, Telecommunication Systems 22 (1) (2003) 267–280.

[24] S. Brands, D. Chaum, Distance-bounding protocols (extended abstract), LNCS EUROCRYPT, 1993.

[25] S. Capkun, M. Cagalj, M. Srivastava, Secure localization with hidden and mobile base stations, IEEE INFOCOM, 2006.

[26] N. Sastry, U. Shankar, D. Wagner, Secure verification of location claims, ACM WiSe Workshop, 2003.

[27] F. Malandrino, C. E. Casetti, C.-F. Chiasserini, M. Fiore, R. S. Yokoyama, C. Borgiattino, A-vip: Anonymous verification and inference of positions in vehicular networks, IEEE INFOCOM, 2013.

[28] B. Yu, C.-Z. Xu, B. Xiao, Detecting sybil attacks in vanets, Parallel and Distributed Computing 73 (6) (2013) 746–756.

[29] J.-H. Song, V. W. Wong, V. C. Leung, Secure Location Verification for Vehicular Ad-Hoc Networks, IEEE GLOBECOM, 2008.

[30] P. Golle, D. Greene, J. Staddon, Detecting and Correcting Malicious Data in VANETs, ACM VANET Workshop, 2004.

[31] Q. Chen, D. Jiang, V. Taliwal, L. Delgrossi, IEEE 802.11 based Vehicular Communication Simulation Design for NS-2, ACM VANET Workshop, 2006.

[32] Dedicated Short Range Communications (DSRC) Home. [Online]. Available: http://www.leearmstrong.com/dsrc/dsrchomeset.htm.