Secure Neighbor Position Discovery in Vehicular Networks

Marco Fiore
INSA Lyon/INRIA, France

Claudio Casetti, Carla-Fabiana Chiasserini
Politecnico di Torino, Italy

Panagiotis Papadimitratos
KTH, Sweden

***Abstract*—In vehicular ad hoc networks, knowledge of neighbor positions is a requirement in a number of important tasks. However, distributed techniques to perform secure neighbor position discovery, suitable for highly mobile ad hoc environments, are missing. In this paper, we address this need by proposing a lightweight distributed protocol that relies only on information exchange among neighbors, without any need of a-priori trustworthy nodes. We present a detailed security analysis of our protocol in presence of one or multiple adversaries, and we evaluate its performance in a realistic vehicular environment.**

I. Introduction

Security in vehicular ad hoc networks (VANETs) is widely regarded as a hot research topic. Privacy, traceability of misbehaving nodes, unauthorized message insertion, fraudulent relaying and insecure neighbor discovery are all aspects of primary importance that need to be addressed. In this paper, we tackle the latter issue, and, specifically, secure verification not only of the presence of neighbors but also of their exact location. Indeed, VANETs are among the most likely candidates to benefit from Secure Neighbor Position Discovery (SNPD) when fine-grained location identification is required, e.g., in the case of congestion charging, traffic monitoring, traffic light prioritization for special vehicles, etc. Critical routing tasks, especially those based on geographic routing, also require that neighboring nodes are reliably identified and localized.

The challenges that an SNPD system must address are multi-faceted: (i) devices running an SNPD need to be able to track their own position and relate it to a common, reliable time reference; (ii) on-demand, real-time knowledge of neighbor positions and identities is needed; (iii) neighboring devices can be faulty or under the control or influence of an adversary, and must be properly detected. While we will assume that the devices are compliant with the first requirement, we focus on designing an SNPD mechanism that addresses the latter two requirements and allows nodes to validate the positions of neighbors within their communication range in a distributed manner.

Note that most of previous work [1], [2], [3], [4] has focused on secure neighbor discovery, which represents a subset of the position discovery problem we target, since our goal is to assess not only the authenticity of would-be neighbors, but also the correctness of their position. Our same objective, i.e., secure neighbor position discovery and verification, has been tackled before in the generic field of ad hoc networks, where nodes mobility is limited or absent. In addition, the solutions proposed for such environments rely on the presence of dedicated mobile or hidden base stations [5], or on the availability of a number of collaborating and trustworthy devices [6]. Finally, we remark that the SNPD problem significantly differs from that of location proving [7], whose goal is to allow a centralized authority to verify the position announced by one specific mobile user.

In this work, instead, we envision a system where nodes act individually but cooperate and leverage the contribution of neighbors to weed out wrong-doers. In such a scenario, we propose a lightweight, distributed, and efficient protocol that enables each node to discover and verify the position of its neighbors. The protocol can be executed by any node, at any point in time, without prior knowledge or assumed trustworthiness of the other nodes that participate. Also, our protocol can sustain high-speed nodes and leverages RF transmissions, since other types of communication require line-of-sight (e.g., infra-red) or have short ranges (e.g., ultra-sound) that make them inappropriate for VANETs.

In the rest of the paper, we first introduce the system and adversary model we adopt. The description of our SNPD protocol and of the tests that are run in order to verify neighbor position claims follow. Finally, we present a security analysis of our approach and performance results derived in a realistic vehicular scenario.

II. System and adversary model

We consider a VANET whose nodes communicate over a two-dimensional plane, using a high-bit-rate data link through an RF interface We refer to a pair of nodes as *communication neighbors* if they can hear each other. For bounding purposes, we also define a *proximity range*, R, and we say that two nodes are *physical neighbors* if their geographical distance is not greater than R.

We assume that each node X knows its own location at any point in time with some maximum error ϵ_p and that it can determine the current time. Also, nodes can perform Time of Flight (ToF)-based RF ranging using one single message transmission, with a maximum error equal to ϵ_r (as discussed in [8] this is a reasonable assumption, although it requires modifications to the current off-the-shelf radio interfaces). Both ϵ_p and ϵ_r are assumed to be equal for all nodes.

Each node has a unique identity, and it carries cryptographic keys that allow it to authenticate messages from other nodes in the network. Although there are various ways to enable authentication, here we require only that authentication of messages is done locally and, to facilitate the presentation, we assume that each node, X, holds its own pair of private and public keys, k_X and K_X respectively, as well as a set of one-time use keys $\{k'_X, K'_X\}$. X can encrypt and decrypt data with its key(s) and the public keys of other nodes; also, it can produce digital signatures with its private key. We assume that the binding between X and K_X can be validated by any node, e.g., with the help of a certificate [9], [10], [11]. Also, note that we do not make any assumption on the nature of the identities employed in our system: in particular, we do not require that a vehicle is bound to a same identity all the time. As a consequence, the scenario we envision is compatible with state-of-the-art techniques to preserve the privacy of mobile users [12].

As for the adversary model, we assume that nodes can either comply with the SNPD protocol or deviate from it; in the latter case, we say they are *faulty* or *adversaries*. Here, we are interested in malicious adversaries that inject messages in order to mislead other nodes about their position. In addition, we focus on *internal*, i.e., they possess the cryptographic keys and credentials of system nodes and, thus, they can fully participate in the protocol execution. However, internal adversaries cannot undetectably modify messages from any other node whose cryptographic keys they do not possess, since they lack the capabilities to compromise keys. We define an internal adversary as *knowledgeable* if, at any point in time, it has knowledge of the current position and identity of its communication neighbors, and as *unknowledgeable* otherwise. Finally, there can be *multiple adversarial nodes* present in the network; in this work, we assume that such adversaries act *independently* of each other.

III. Secure neighbor position discovery protocol

The SNPD protocol we propose allows any node in the network to discover and verify the position of its *communication neighbors* that participate in the protocol message exchange. The procedure is performed in a reactive manner, i.e., it can be run by any node at any time instant, by initiating the message exchange. Such node will be referred to as the *verifier*.

Our solution is based on a best effort, cooperative approach. It aims at verifying the position only of the neighbors with which the message exchange takes place successfully. It therefore disregards nodes for which the protocol exchange prematurely ends, e.g., due to message losses on the channel, or communication neighbors that refuse to take part in the protocol. The scheme assumes that the node position does not vary significantly during the message exchange, as confirmed by simulation results[1].

The SNPD protocol leverages the information collected by neighboring nodes thanks to the broadcast nature of the wireless medium. Such information, fed back to the verifier, is used to compute, via ToF-based ranging, the distance between pairs of neighbors. Based on this knowledge, the verifier performs security tests to tag its communication neighbors as:

- *verified*, i.e., nodes the verifier deems to be trustworthy;
- *faulty*, i.e., nodes the verifier deems to have announced an incorrect position;
- *unverifiable*, i.e., nodes the verifier cannot prove to be either correct or faulty – this may happen due to lack of sufficient information on these nodes or because the verifier cannot form a clear opinion on their behavior.

Clearly, the objective of our SNPD protocol is to be robust to adversarial nodes in that it minimizes the number of unverifiable nodes and the number of positive/negative falses. By the latter we mean correct nodes declared as faulty and adversaries tagged as verified.

Below, we detail the message exchange between the verifier and its communication neighbors, followed by a description of the security tests run by the verifier.

A. Message exchange

Let t_X be the time at which a node X starts a broadcast transmission, and t_{XY} the time at which a node Y starts receiving that same transmission; p_X is the current position of X, and $\mathcal{N}_X$ is the current set of its communication neighbors.

Consider a verifier S that initiates the SNPD protocol. The message exchange procedure is outlined in Algorithm 1 for S, and in Algorithm 2 for any of S's communication neighbors.

Algorithm 1: Message exchange protocol: verifier node

```
1  node $S$ do
2      $S \rightarrow * : \langle \text{POLL}, K'_S \rangle$
3      $S$ : store $t_S$
4      when receive REPLY from $Y \in \mathcal{N}_S$ do
5          $S$ : store $t_{YS}, c_Y$
6      end
7      after $T_{max} + \Delta + T_{jitter}$ do
8          $S \rightarrow * : \langle \text{REVEAL}, E_{k'_S}\{h_{K'_S}\}, K_S, Sig_S \rangle$
9      end
10 end
```

The verifier starts the protocol by broadcasting a POLL whose transmission time t_S is stored locally (Alg. 1, lines 2-3). Such message is anonymous, since (i) it does not contain the verifier's identity, (ii) it is transmitted employing a fresh MAC address, and (iii) it contains a public key K'_S from a one-time use private/public key pair k'_S, K'_S, taken from a pool of anonymous keys which do not allow neighbors to map them onto a specific node. Note that including a one-time key in the the POLL also ensures that the message is fresh. Furthermore, including the public key in broadcast messages makes the protocol self-contained, as it does not have to rely on a separate asymmetric key exchange; as for key management, it can exploit one of the architectures proposed in the literature, e.g., [10].

[1] In an overcrowded scenario featuring 50 neighbors moving at an average speed of about 30 km/h, the average duration of the message exchange spanned over 150 ms, resulting in an average of 10% of colliding nodes; such a time interval corresponds to an average position shift of 1.2 m.

Algorithm 2: Message exchange protocol: neighbor node

```
1  forall $X \in \mathcal{N}_S$ do
2      when receive POLL by $S$ do
3          $X$ : store $t_{SX}$
4          $X$ : extract $T_X$ uniform r.v. $\in [0, T_{max}]$
5      end
6      after $T_X$ do
7          $X : c_X = E_{K'_S}\{t_{SX}, K_X, Sig_X\}$
8          $X \to * : \langle \text{REPLY}, c_X, h_{K'_S} \rangle$
9          $X$ : store $t_X$
10     end
11     when receive REPLY from $Y \in \mathcal{N}_S \cap \mathcal{N}_X$ do
12         $X$ : store $t_{YX}, c_Y$
13     end
14     when receive REVEAL from $S$ do
15         $X : \tau_X = \{(t_{YX}, c_Y)\ \forall Y \in \mathcal{N}_S \cap \mathcal{N}_X\}$
16         $X \to S : \langle \text{REPORT}, E_{K_S}\{p_X, t_X, \tau_X, Sig_X\} \rangle$
17     end
18 end
```

A generic communication neighbor $X \in \mathcal{N}_S$ that receives the POLL stores its reception time t_{SX}, and extracts a random wait interval $T_X \in [0, T_{max}]$ (Alg. 2, lines 2-5). After T_X has elapsed, X broadcasts a REPLY message using a fresh MAC address, and records the corresponding transmission time t_X (Alg. 2, lines 6-10). The REPLY contains encrypted information for S, namely the signed neighbor identity, Sig_X, and the POLL reception time: we refer to these data as X's *commitment* and tag it as c_X. The hash $h_{K'_S}$, derived from the verifier's public key, K'_S, is also included to bind POLL and REPLY belonging to the same message exchange.

Upon reception of a REPLY message from a communication neighbor Y, the verifier S stores the reception time t_{YS} and the commitment c_Y (Alg. 1, lines 4-6). A different communication neighbor X receives the REPLY message broadcast by Y, if Y is a communication neighbor of both S and X, i.e., $Y \in \mathcal{N}_S \cap \mathcal{N}_X$. In such case, X too stores the reception time t_{YX} and the commitment c_Y (Alg. 2, lines 11-13). Note that also REPLY messages are anonymous, hence a node records all commitments it receives without knowing their origin.

After a time $T_{max} + \Delta + T_{jitter}$, S broadcasts a REVEAL message; Δ accounts for the propagation and contention lag of REPLY messages scheduled at time T_{max}, and T_{jitter} is a random time added to thwart jamming efforts on this message. Through the REVEAL, (i) S unveils its identity by including its signature and its public key to decrypt it, and (ii) it proves to be the author of the original POLL. The latter is achieved by attaching the encrypted hash $E_{k'_S}\{h_{K'_S}\}$ (Alg. 1, lines 7-9).

Once the identity of the verifier is known, each neighbor X, which received S's original POLL, unicasts to S an encrypted and signed REPORT message containing its own position, the transmission time of its REPLY, and the list of pairs of reception times and commitments referring to the REPLY broadcasts it received (Alg. 2, lines 14-17). Commitments are included 'as they are', since only S can decrypt them and match the identity of the nodes that created the commitments with the reported reception times. We also point out that, by transmitting its own position only after the reception of the REVEAL, a neighbor prevents a verifier from exploiting anonymity to run a flooding attack.

B. Position verification

Once the message exchange is concluded, S decrypts the received data and acquires the position of all neighbors that participated in the protocol, i.e., p_X, $\forall X \in \mathcal{N}_S$. S also knows the transmission time of its POLL and learns the transmission time of all subsequent REPLY messages, as well as the corresponding reception times recorded by the recipients of such broadcasts. Applying a ToF-based technique, S can thus compute its distance from each communication neighbor, as well as the distances between pairs of communication neighbors that happen to share a link. In particular, denoting by c the speed of light, we define $d_{XY} = (t_{XY} - t_X) \cdot c$, i.e., the distance that S computes from the timing information it collected about the broadcast message sent by X. Similarly, we define $d_{YX} = (t_{YX} - t_Y) \cdot c$, i.e., the distance that S computes using the information related to the broadcast by Y. Exploiting its knowledge, the verifier can run verification tests to fill the set $\mathcal{F}_S$ of faulty communication neighbors, the set $\mathcal{V}_S$ of verified nodes, and the unverifiable set $\mathcal{U}_S$.

The first verification is the *Direct Symmetry* (DS) test, detailed in Algorithm 3. For direct links between the verifier and each of its communication neighbors, S checks whether reciprocal ToF-derived distances are consistent (i) with each other, (ii) with the position advertised by the neighbor, and (iii) with the proximity range R. To this end, we denote by $|x|$ the modulus of x and by $\|p_X - p_Y\|$ the Euclidean distance between locations p_X and p_Y. The first check is performed by comparing the distances d_{SX} and d_{XS} obtained from ranging, which shall not differ by more than twice the ranging error (line 4). The second check verifies that the position advertised by the neighbor is consistent with such distances, within an error margin equal to $2\epsilon_p + \epsilon_r$ (line 5). This check is trivial but fundamental, since it correlates positions to verified distances: without it, an attacker could fool the verifier by simply advertising an arbitrary position along with correct broadcast transmission and reception timings. Finally, S verifies that d_{SX} is not larger than R (line 6), and declares

Algorithm 3: Direct Symmetry (DS) test

```
1  node $S$ do
2      $S : \mathcal{F}_S \leftarrow \emptyset$
3      forall $X \in \mathcal{N}_S$ do
4          if $|d_{SX} - d_{XS}| > 2\epsilon_r$ or
5             $|\|p_S - p_X\| - d_{SX}| > 2\epsilon_p + \epsilon_r$ or
6             $d_{SX} > R$ then
7              $S : \mathcal{F}_S \leftarrow X$
8          endif
9      end
10 end
```

a neighbor as faulty if a mismatch surfaced in any of these checks[2].

The DS test implies *direct* verifications that compare trusted information collected by the verifier against data advertised by each neighbor. The content of the messages received by S, however, allows also *cross*-verifications, i.e., checks on the information *mutually* gathered by each pair of communicating neighbors. Such checks are done in the *Cross-Symmetry* (CS) test, in Algorithm 4.

In particular, the CS test excludes nodes already declared as faulty by the DS test (line 6) and only considers nodes that proved to be communication neighbors between each other, i.e., for which ToF-derived mutual distances are available (line 7). Then, it verifies the symmetry of such distances (line 9), their consistency with the positions declared by the nodes (line 10), and their feasibility with respect to the proximity range (line 11). For each communication neighbor X, a link counter l_X and a mismatch counter m_X are maintained. The former is incremented at every new cross-verification on X, and records the number of links between X and other communication neighbors of S (line 8). The latter is incremented every time at least one of the cross-checks on distances and positions fails (line 12), and identifies the potential for X being faulty.

Algorithm 4: Cross-Symmetry (CS) test

```
1  node $S$ do
2      $S : \mathcal{U}_S \leftarrow \emptyset$, $\mathcal{V}_S \leftarrow \emptyset$
3      forall $X \in \mathcal{N}_S,\ X \notin \mathcal{F}_S$ do
4          $S : l_X = 0,\ m_X = 0$
5      end
6      forall $(X,Y) \mid X,Y \in \mathcal{N}_S,\ X,Y \notin \mathcal{F}_S,\ X \neq Y$ do
7          if $\exists\, d_{XY}, d_{YX}$ then
8              $S : l_X = l_X + 1,\ l_Y = l_Y + 1$
9              if $|d_{XY} - d_{YX}| > 2\epsilon_r$ or
10                 $|\|p_X - p_Y\| - d_{XY}| > 2\epsilon_p + \epsilon_r$ or
11                 $d_{XY} > R$ then
12                 $S : m_X = m_X + 1,\ m_Y = m_Y + 1$
13             endif
14         end
15     end
16     forall $X \in \mathcal{N}_S,\ X \notin \mathcal{F}_S$ do
17         if $l_X < 2$ then  $S : \mathcal{U}_S \leftarrow X$
18         else  switch $\frac{m_X}{l_X}$ do
19             case $\frac{m_X}{l_X} > \delta$  $S : \mathcal{F}_S \leftarrow X$
20             case $\frac{m_X}{l_X} = \delta$  $S : \mathcal{U}_S \leftarrow X$
21             case $\frac{m_X}{l_X} < \delta$  $S : \mathcal{V}_S \leftarrow X$
22         end
23     end
24 end
```

Once all neighbor pairs have been processed, a node X is added to the unverifiable set $\mathcal{U}_S$ if it shares less than two neighbors with S (line 17). Indeed, in this case the information available on the node is considered to be insufficient to tag the node as verified or faulty (see Section IV for more details). Otherwise, if S and X have two or more common neighbors, X is declared as faulty, unverifiable, or verified, depending on the mismatch percentage of cross-checks it was involved in (lines 18-22). More precisely, X is added to $\mathcal{F}_S$, $\mathcal{U}_S$ or $\mathcal{V}_S$, depending on whether the ratio of the number of mismatches to the number of checks is greater than, equal to, or less than δ.

We point out that the lower the δ, the fewer the failed cross-checks needed to declare a node as faulty, while the higher the δ, the higher the probability of false negatives. In the following, we set $\delta = 0.5$ so that a majority rule is enforced: the verifier makes a decision on the correctness of a node by relying on the opinion of the majority of shared communication neighbors. If not enough common neighbors are available to build a reliable majority, the node is unverifiable. As shown in the next section, this choice makes our SNPD protocol robust to attacks in many different situations.

IV. Security analysis

The message exchange and tests described above have several security implications that deserve to be stressed.

i) POLL and REPLY are anonymous: an adversary can know neither who the verifier is, nor which neighbor sent which REPLY. Thus, a knowledgeable adversary can exploit the information it has on its neighbors only if it can guess correctly the identity of the POLL and REPLY senders. The larger the neighborhood, the lower the probability of a correct guess.

ii) The verifier discloses its identity with the REVEAL only after its neighbors have sent a committing information on the time at which they received the original POLL. In other words, an attacker has already advertised such time information when it knows who the verifier is, and thus has committed itself to a position it cannot modify.

iii) All messages are signed by the source nodes, guaranteeing their authenticity. Also, all information on node positions is unicast to the verifier and encrypted with its public key, which ensures that the verifier is the only one capable of decrypting it and no positioning information is leaked in the network.

iv) There is no pre-established sequence in the transmission of REPLY or REPORT messages by nodes; rather, the unpredictability of their transmission by a given neighbor makes the protocol robust to selective jamming attacks: if an adversary wants to harm the reception at S of a REPLY or REPORT sent by a neighbor, it has to jam the medium for almost the whole duration of the SNPD message exchange.

v) If a node M rebroadcasts the POLL and relays the subsequent REPLY received from a correct node A, which is not a communication neighbor of S (replay attack), it cannot trick S into believing that A is its neighbor. Indeed, M cannot tamper with the timing t_{SA}, hence the value of d_{SA} will reveal (in the DS test) that A is too far away from S to be a communication or even a physical neighbor. For similar reasons, a wormhole attack is also destined to failure.

vi) An adversary can disregard the REPLY messages sent by some neighbors it shares with S, and exclude from its REPORT the commitments received from them (we call this REPLY disregard attack). It can therefore reduce its advertised

[2] In the algorithm the 2nd and 3rd checks are performed on d_{SX} only, for clarity of presentation; the same checks should be done on d_{XS}. This, however, has no implication on the following security analysis of the protocol.

neighborhood to be a subset of the actual one, and avoid to be cross-checked with correct nodes during the CS test. However, the node will end up being tagged at the best as unverifiable and no gain will be achieved.

The above observations are instrumental to the analysis of several scenarios in which we test the robustness of our solution. In particular, in Sections IV-A and IV-B we motivate our choice to tag a node as unverifiable when, in the CS test, it shares at most one communication neighbor with the verifier. In Section IV-C, we consider that the verifier shares two or more communication neighbors with an adversary, and we show that, in this case, the adversary is very likely declared as faulty. Note that the above scenarios may represent either the actual topology of the neighborhood shared between the verifier and an adversarial node, or the topology resulting from a REPLY disregard attack. Finally, in Section IV-D we discuss the case of multiple independent adversaries.

A. Single adversary, no common neighbors

Consider a verifier S that has an adversary M and no common neighbors between them. In order to bring a successful attack, M must tamper with the data S uses for ranging, so that the resulting distance confirms its fake advertised position. To this end, M can forge at its convenience the time information in the messages it generates. In particular, let p'_M be the fake position that M wants to advertise; we denote by t'_{SM} the fake timing that M introduces in its REPLY, and by t'_M the fake timing inserted in its REPORT (in addition to p'_M).

The DS test (Alg. 3) run by S on M checks the consistency between distances, by verifying that $|d_{SM} - d_{MS}| \le 2\epsilon_r$, or:

$$|(t'_{SM} - t_S) \cdot c - (t_{MS} - t'_M) \cdot c| \le 2\epsilon_r \tag{1}$$

and that positions are also coherent with the distances, i.e., $|\|p_S - p'_M\| - d_{SM}| \le 2\epsilon_p + \epsilon_r$, or, equivalently:

$$|\|p_S - p'_M\| - (t'_{SM} - t_S) \cdot c| \le 2\epsilon_p + \epsilon_r \tag{2}$$

Thus, the adversary must forge t'_M and t'_{SM}, so that (1)–(2) still hold after its real position p_M is replaced with p'_M.

Solving the equation system obtained by setting the error margin to zero in (1)–(2), and expressing the ToF using the node positions, we obtain:

$$\begin{aligned} t'_M &= t_{MS} - \frac{\|p_S - p'_M\|}{c} \\ &= t_M + \frac{\|p_S - p_M\|}{c} - \frac{\|p_S - p'_M\|}{c} \end{aligned} \tag{3}$$

$$\begin{aligned} t'_{SM} &= t_S + \frac{\|p_S - p'_M\|}{c} \\ &= t_{SM} - \frac{\|p_S - p_M\|}{c} + \frac{\|p_S - p'_M\|}{c} \end{aligned} \tag{4}$$

Note that p'_M is chosen by M, and that M knows t_M in (3) (since this is the actual transmission time of its own REPLY) and t_{SM} in (4) (since this is the time at which it actually received the POLL from S). We therefore have a system of two equations that M can solve in the two unknowns t'_M and t'_{SM} if it is aware of p_S (knowledgeable adversary). We stress that M can be knowledgeable only: (i) if it has previously run the SNPD protocol to discover the position and identity of its neighbors, and (ii) if the verifier's position has not changed since such discovery procedure. As M cannot foresee when S starts the SNPD protocol, such conditions are extremely hard to fulfill, especially in highly dynamic environments.

Nevertheless, if M is aware of S's location, the advertised position p'_M will pass the DS test provided that it is within the proximity range R. Given such potential weakness, the SNPD protocol marks isolated neighbors as unverifiable in the CS test, even if they pass the DS test.

B. Single adversary, one common neighbor

We now add to the previous scenario a node X, which is a correct neighbor common to S and M. Recall that, in bringing its attack, M can forge messages with altered information, but it cannot modify the content of messages sent by others, since they are all encrypted and signed.

The discussion in Section IV-A applies again, since the fake position advertised by M needs to pass the DS test: M must be aware of S's current position and must forge t'_M and t'_{SM} according to p_S and p'_M. However, the presence of the common neighbor allows two additional levels of security.

First, the POLL and REPLY messages are anonymous, hence M does not know if the protocol verifier is S or X upon reception of such messages. Also, if it wants to take part in the protocol, M is forced to advertise a committing information (namely, the POLL reception time, t_{SM}) in its REPLY, before receiving the REVEAL and discovering the verifier's identity. The only option for M is then to *randomly guess* who the verifier is, and properly change t_{SM} into t'_{SM}, as in (4): this leads to a 0.5 probability of failure in the attack.

Second, the CS test run by S on the pair (M, X), requires that $|d_{XM} - d_{MX}| \le 2\epsilon_r$, and $|\|p_X - p_M\| - d_{XM}| \le 2\epsilon_p + \epsilon_r$. As shown earlier for the DS test, to pass these checks, M is forced to advertise the following fake timings

$$t'_M = t_M + \frac{\|p_X - p_M\|}{c} - \frac{\|p_X - p'_M\|}{c} \tag{5}$$

$$t'_{XM} = t_{XM} - \frac{\|p_X - p_M\|}{c} + \frac{\|p_X - p'_M\|}{c} \tag{6}$$

Clearly, M needs to know X's current position, p_X, to solve (5) and (6), thus it must be a knowledgeable adversary. Moreover, while the forged t'_{XM} computed in (6) can be easily announced by M in its REPORT to S, the adversary has now two expressions for t'_M, in (3) and (5). Since it can only advertise one value of t'_M, the only chance for M to pass both the DS and CS tests is that the values returned by (3) and (5) coincide, which implies

$$\|p_S - p_M\| - \|p_S - p'_M\| = \|p_X - p_M\| - \|p_X - p'_M\| \tag{7}$$

In other words, M is constrained to choose locations with the same distance increment (or decrement) from S and X. In (7), p_S, p_X, and p_M are fixed and known, hence distances between p_S and p_M, and between p_X and p_M can be considered as constant. Since p'_M is variable over the plane, we rewrite (7) as $\|p_X - p'_M\| - \|p_S - p'_M\| = k$, which is

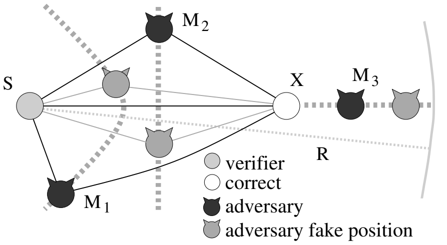

Fig. 1. M_1, M_2, and M_3 depict different situations in which a single adversary can be. In the general case (as M_1), a knowledgeable adversary that correctly guessed the verifier's identity can pass all tests if its fake position is on a hyperbola with foci in S, X, passing by M_1. Particular cases are: (i) the adversary is equidistant from S and X (as M_2), constraining the fake position on the symmetry axis of S and X; (ii) the adversary is aligned with S and X (as M_3), and not between them: then, the fake location needs to be on the same line, between X and a point at distance R from S.

the equation describing a hyperbola with foci in p_S and p_X, and passing through p_M. It follows that only positions on such hyperbola satisfy the four constraints in (3), (4), (5), and (6), and p'_M must lie on that curve in order to pass all tests. Examples of this condition are shown in Figure 1.

Summarizing, the presence of a common neighbor X drastically reduces the vulnerability of the verifier to attacks, since M is now required (i) to be knowledgeable, (ii) to correctly guess the verifier's identity, and (iii) to advertise a fake position only along a specific curve. However, since some space for successful attacks remains, the CS test marks as unverifiable nodes that passed the DS test but share only one neighbor with the verifier. We also stress that, if M tweaks the timings so as to pass the DS test and does not care about the matching with X, it will still be tagged as unverifiable.

C. Single adversary, two or more common neighbors

In the case of two or more common neighbors, we split the discussion in the two following cases.

Generic network topology. When a second correct neighbor Y is shared between S and M, the discussion in Section IV-B can be extended as follows [3].

As before, the adversary M has to be knowledgeable, but the presence of a second common neighbor reduces to 0.33 the probability that M correctly guesses the identity of the verifier. More importantly, by applying the same reasoning as in Section IV-B, M has now to forge four time values, i.e., t'_M, t'_{SM}, t'_{XM}, and t'_{YM}, so that six equations are satisfied, i.e., (3), (4), (5), (6), and the two equations corresponding to the cross-check with the second common neighbor Y [4].

To fulfill the constraints on t'_M, now M has to announce a position p'_M that is equally farther from (or closer to) S, X and Y with respect to its actual location p_M. The point satisfying such condition lies at the intersection of three hyperbolae with foci in p_S and p_X, p_S and p_Y, p_X and p_Y, respectively, and that point must necessarily be the real position of the adversary, p_M.

TABLE I
SUMMARY OF SECURITY ANALYSIS IN GENERIC NETWORK TOPOLOGY

$\lvert\mathcal{N}_S \setminus X\rvert$ X	0	1	2	3+
Correct	$\mathcal{U}_S$	$\mathcal{U}_S$	$\mathcal{V}_S$	$\mathcal{V}_S$
Unknowledgeable adv.	$\mathcal{F}_S$	$\mathcal{F}_S$	$\mathcal{F}_S$	$\mathcal{F}_S$
Knowledgeable adv.	$\mathcal{U}_S$	$\mathcal{U}_S$ (0.5) $\mathcal{F}_S$ (0.5)	$\mathcal{U}_S$ (0.165) $\mathcal{F}_S$ (0.835)	$\mathcal{F}_S$

Accordingly, in presence of two common neighbors, the CS test marks a node with no mismatches as verified. The majority rule (i.e., $\delta = 0.5$) results instead in the adversary being tagged as faulty when mismatches are recorded with both common neighbors. Finally, the adversary is added to the unverifiable set, if it is capable of fooling S and either X or Y, since that leads to one mismatch over two links checked.

We stress that deceiving S and one of the common neighbors requires, beside the knowledge of their current positions and a correct guess on the verifier's identity, also the pinning of which REPLY comes from which neighbor (i.e., M must randomly map t_{XM} onto p_X and t_{YM} onto p_Y for the computations on the hyperbolae to work). Thus, the guess taken by M in the hope of being marked as unverifiable has a success probability jointly given by the probability of guessing the right verifier (0.33) and the probability of guessing the right mapping (0.5) of REPLY reception times onto neighbor positions, which yields 0.165.

Finally, the presence of three or more common neighbors between S and M totally impairs the chances of attacks. Indeed, not only does the probability of guessing the right originators of the different messages shrink as the size of the common neighborhood grows, but the majority rule dooms the adversary to insertion in the faulty set, even when all random guesses are exact. By extending the above analysis on the hyperbolae, we observe that, when S and M share $n \geq 3$ communication neighbors, the number of mismatches found in the CS test is at least $n-1$, implying a mismatch-to-links ratio always greater than $\delta = 0.5$.

A summary of the security of the SNPD protocol, in presence of a single adversary and in a generic network topology, is presented in Tab. I, where different rows identify different behaviors of the neighbor X under verification by S. The columns represent the number of neighbors, assumed to be correct, shared by S and X. For each combination, we report the set to which X is assigned by S, possibly with a probability value due to the random guessing on neighbors' roles by the adversary.

Collinear nodes. When the majority of common neighbors is collinear to S and an adversary M, and lies on the same side as S with respect to p_M, a degree of freedom exists for the attacker. Indeed, M is verified if it announces a fake position that is collinear with p_M and p_S, within a distance R from S, and such that the majority of the common neighbors still lies on the same side as S with respect to p'_M. This case, however, hardly brings any gain to the adversary, since M

[3] We do not make any assumption on the connectivity between X and Y.

[4] The latter two equations can be obtained from (5)–(6) by replacing p_X, t_{XM} and t'_{XM}, respectively, with p_Y, t_{YM} and t'_{YM}.

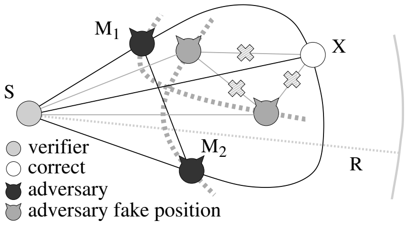

Fig. 2. Clique of four nodes: the verifier S, a correct neighbor X, and two adversaries (M_1, M_2). M_1 (M_2) announces a fake position along a hyperbola with foci on p_S and p'_{M_2} (p'_{M_1}). However, the latter information is fake, leading to a mismatch in the cross-check on (M_1,M_2). Also, since each attacker can "move" at most one link other than that with S, the checks on (X,M_1) and (X,M_2) fail as well. Thus, M_1 and M_2 damage each other and are tagged as faulty. X, although correct, is added to $\mathcal{F}_S$, since all neighbors it shares with S happen to be adversaries.

must remain aligned with the other nodes, cannot change its order with respect to the majority of them, and has to be within S's proximity range.

D. Multiple adversaries

Let us now consider that multiple independent adversaries $M_1, \ldots, M_n$, are communication neighbors of the verifier.

The adversaries cannot be of any benefit to each other. Indeed, each of the M_i nodes announces a false own position but it is unaware of the presence of the other adversaries, let alone of their false position claims and their replies to S. As a result, the reception times that M_i provides to S are inconsistent with the faulty claims of M_j, $j \neq i$. This increases the mismatch count in the CS test, for each adversary. In short, multiple independent attackers add to the mismatch count due to correct nodes' announcements and make it even more likely that the M_i nodes are tagged as faulty. Of course, multiple adversaries may also affect the classification of correct nodes: due to the majority rule, a correct node is tagged as faulty (unverifiable) if it shares with S a number of adversaries greater than (equal to) the number of correct nodes. An example of faulty marking of a correct node is shown in Figure 2.

V. Performance evaluation

We test our SNPD protocol in a real-world road topology that consists of a 5×5 km^2 portion of the urban area of the city of Zurich [13]. These traces describe the individual movement of cars through a queue-based model calibrated on real data: they thus provide a realistic representation of vehicular mobility at both microscopic and macroscopic levels [14]. We extracted 3 hours of vehicular mobility, in presence of mild to heavy traffic density conditions; the average number of cars in the area at a given time is 1200.

The trace has a time discretization of 1 s. Thus, every second we randomly select 1% of the nodes as verifiers. For each node, we consider that all devices within the proximity range R are communication neighbors of the node, and that transmissions between communication neighbors are always successful. This assumption is justified by the link duration we extracted from the traces, e.g., around 10 s, which is significantly longer than the message exchange duration observed through simulations at the network level (omitted for lack of space). Clearly, the larger the R, the higher the number of neighbors taking part in the same instance of the SNPD protocol: for example, for R equal to 50 m and 500 m, the average node degree is 8 and 104.8 and the variance is 5.9 and 71.8, respectively. Also, we set ϵ_r to 6.8 m, and ϵ_p to 5 m and 1 m for cars and pedestrian nodes, respectively [8].

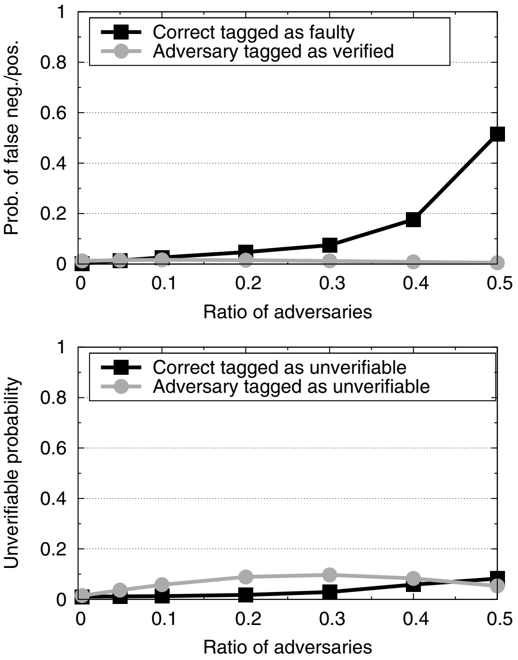

Fig. 3. Probability of false negatives/positives (top) and probability of classifying a neighbor as unverifiable (bottom), for $R = 250$ m and as the ratio of adversaries varies.

Since unknowledgeable adversaries are always tagged as faulty in the DS test, in the following we present results considering that all adversaries are *knowledgeable*. We stress that this is a very hard condition to meet in dynamic networks, hence all results are to be considered as an upper bound to the success probability of an attack. In our simulation tests, we randomly select a ratio (a varying parameter in our analysis) of the nodes as attackers. For every scenario under study, we statistically quantify the outcome of the verification test and compare it to the actual behavioral model of the nodes (namely, correct or adversary).

Before discussing the results, we remark a fundamental aspect of the addressed environment. The platooning of vehicles on a road gives adversaries the chance to deploy themselves on a straight line where the verifier and two common neighbors also lie. As explained in Section IV-C, this condition preludes to a successful attack.

The results in terms of the probabilities that the tests return false positives and false negatives are presented in the top plots in Figures 3 and 4, while the probabilities that a (correct or adversarial) node is tagged as unverifiable are shown in the

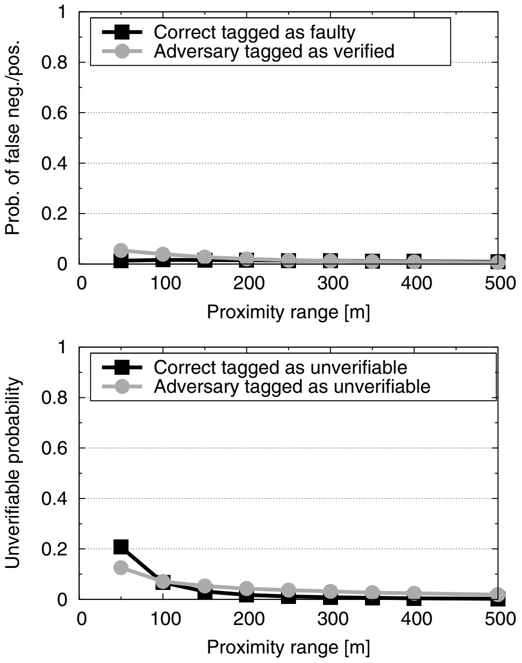

Fig. 4. Probability of false negatives/positives (top) and probability of classifying a neighbor as unverifiable (bottom), when the ratio of adversaries is 0.05 and the proximity range R varies.

bottom plots. The former gauge the reliability of our scheme, while the latter are a mark of the protocol accuracy. The plots showing the false positives and false negatives, when the ratio of adversaries varies and $R = 250$ m, confirm that our scheme errs on the side of caution: indeed, as the number of adversaries increases, it is more likely for a correct node to be mislabeled than for an adversary to be verified (the latter probability amounting to less than 0.02). Instead, widening the proximity range with a fixed adversary ratio, namely 0.05, only plays into the verifier's hands, thanks to the greater number of nodes (the majority of which are correct) that can be tested. As for the probability that a node is unverifiable, while little sensitivity to the ratio of adversaries is observed, a small R (hence fewer neighbors) affects the protocol capability to reach a conclusive verdict on either correct or adversarial nodes. We also estimated that the degree of freedom that an adversary has in setting its fake position, for $R = 250$ m and a ratio of 0.05 attackers, is such that, on average, the fake and actual positions of a verified adversary are collinear and differ by 40 m.

VI. Conclusion

We proposed a lightweight, distributed scheme for securely discovering the position of communication neighbors in vehicular ad hoc networks. Our solution does not require the use of a-priori trustworthy nodes, but it leverages the information exchange between neighbors. Although simple, our analysis showed the scheme to be very effective in identifying adversarial nodes. Results derived using realistic vehicular traces confirmed such ability and highlighted the good performance of our solution in terms of both false negatives/positives and uncertain neighbor classifications.

Acknowledgment

This work was supported by Regione Piemonte through the MASP project.

References

[1] S. Brands, D. Chaum, "Distance Bounding Protocols," *EUROCRYPT*, 1993.

[2] P. Papadimitratos, M. Poturalski, P. Schaller, P. Lafourcade, D. Basin, S. Čapkun, J.-P. Hubaux, "Secure Neighborhood Discovery: A Fundamental Element for Mobile Ad Hoc Networking," *IEEE Comm. Mag.*, vol. 46, no. 2, Feb. 2008.

[3] M. Poturalski, P. Papadimitratos, J-P. Hubaux, "Secure Neighbor Discovery in Wireless Networks: Formal Investigation of Possibility," *ASIACCS*, 2008.

[4] M. Poturalksi, P. Papadimitratos, J.-P. Hubaux, "Towards Provable Secure Neighbor Discovery in Wireless Networks," *Workshop on Formal Methods in Security Engineering*, Alexandria, VA, Oct. 2008.

[5] S. Capkun, K. Rasmussen, M. Cagalj, M. Srivastava, "Secure Location Verification with Hidden and Mobile Base Stations," *IEEE Trans. on Mobile Comp.*, 2008.

[6] S. Čapkun, J.-P. Hubaux, "Secure Positioning in Wireless Networks," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 24, no. 2, pp. 221–232, 2006.

[7] Z. Zhu, G. Cao, "APPLAUS: A Privacy-Preserving Location Proof Updating System for Location-based Services," *IEEE Infocom*, 2011.

[8] M. Fiore, C. Casetti, C.-F. Chiasserini, P. Papadimitratos, "SNPD Protocol: Security Analysis and Implementation Issues," *Tech. Rep.,* Politecnico di Torino, July 2009, www1.tlc.polito.it/casetti/Techrep0709.pdf.

[9] P. Papadimitratos, L. Buttya, T. Holczer, E. Schoch, J. Freudiger, M. Raya, Z. Ma, F. Kargl, A. Kung, J.-P. Hubaux, "Secure Vehicular Communication Systems: Design and Architecture," *IEEE Communcations Magazine*, vol. 46, no. 11, pp. 100–109, 2008.

[10] A. Studer, E. Shi, F. Bai, A. Perrig, "TACKing Together Efficient Authentication, Revocation, and Privacy in VANETs," *Secon*, 2009.

[11] G. Calandriello, P. Papadimitratos, J.P. Hubaux, A. Lioy, "On the Performance of Secure Vehicular Communication Systems," *IEEE Transactions on Dependable and Secure Computing*, to appear.

[12] D. Eckhoff, C. Sommer, T. Gansen, R. German, F. Dressler, "Strong and Affordable Location Privacy in VANETs: Identity Diffusion Using Time-Slots and Swapping," *IEEE VNC*, 2010.

[13] ETH traces, http://lst.inf.ethz.ch/ad-hoc/car-traces.

[14] N. Cetin, A. Burri, K. Nagel, "A Large-scale Multi-agent Traffic Microsimulation Based on Queue Model," *STRC*, 2003.