Sybil-Resistant Meta Strategies for the Forwarder's Dilemma

Yunus Durmus Andreas Loukas Koen Langendoen
Embedded Software Group
Delft University of Technology
E-mail: {y.durmus, a.loukas, k.g.langendoen}@tudelft.nl

Ertan Onur
Department of Computer Engineering
Middle East Technical University
E-mail: eronur@metu.edu.tr

***Abstract*—Cooperation is the foundation of wireless ad hoc networks with nodes forwarding their neighbors' packets for the common good. However, energy and bandwidth constraints combined with selfish behavior lead to collapsed networks where all nodes defect. Researchers have tried to incentivize or enforce the nodes for cooperation in various ways. However, these techniques do not consider the heterogeneous networks in which a diverse set of nodes with different cognitive capabilities exist. Furthermore, in ad hoc networks identity is a fuzzy concept. It is easy to forge multiple identities and hide defective behavior. Moreover, the nature of the wireless medium is always ambiguous due to collisions, interference and asymmetric links. In all this uncertainty, having complete information about the intentions of the nodes and acting on it is not straightforward. Backed by evolutionary game theory and multi-agent systems research, we adapt and modify two meta strategies to embrace this uncertainty. These modified meta strategies, Win Stay Loose Shift and Stochastic Imitate Best Strategy, do not require strict identity information and only depend on nodes' own capabilities. Nodes monitor the traffic in their neighborhood by using a two-hop overhearing method, and decide whether they should be cooperative or defective. We show that nodes are able to discover and use the best strategy in their locality and protect themselves against the exploitation by free riders who devise Sybil attacks by changing their identities.**

Keywords*—*Wireless Ad Hoc Networks, Forwarder's Dilemma, Cooperation, Evolutionary Game Theory, Network Reciprocity, Win Stay Loose Shift, Imitate Best Strategy.

I. Introduction

In ad hoc networks, multi-hop communication is only possible when devices cooperate. After all an injected packet can only reach its destination if it is relayed (forwarded) by other devices. Hence, cooperation is at the foundation of wireless ad hoc networks. Unfortunately, in the absence of a central authority, the rational behavior is to be selfish; rationality implies maximizing the utility of one's actions, which undermines cooperation. In the case of ad hoc networks consisting of utility-maximizing energy-constrained devices, the rational behavior is to defect and drop packets rather than to cooperate and forward them, saving precious energy doing so. This strategic situation of being unwilling to relay packets of other nodes, yet requiring the help of other nodes to forward ones own packets, is referred to as the *forwarder's dilemma*. It is a kind of well-known game called prisoner's dilemma in classical game theory where the equilibrium is defection. As a consequence, regarding the classical game theory, multi-hop communication in an ad hoc network is impossible when cooperation is not enforced.

In the last decade, researchers have aimed at incentivizing or enforcing cooperation with methods that incorporate rewards and punishments such as *credit exchange*, *direct-* and *indirect reciprocity*. This has proven to be difficult for two reasons. First, the scope of these methods is limited due to the strong, underlying assumptions such as the availability of tamper-proof hardware [2] and strong encryption (see Section VII). Second, these methods share a common Achilles heel: they depend on *identity* information, making them susceptible to Sybil attacks where malicious nodes fabricate and spawn identities at will. Adopting a new identity allows attackers to, for example, whitewash (clear) their reputation as a free rider by pretending to freshly join the network. Note that these free riders do not employ a Sybil attack with a malicious intent, i.e. they do not subvert the network, as that would not allow them to abuse the generosity of cooperative nodes. Slandering (good/bad mouthing) is another type of Sybil attack, in which an attacker credits a false rumor by impersonating multiple (fake) sources. *Indirect reciprocity* is especially prone to slandering. Although many proposals for avoiding Sybil attacks have been proposed, including the use of trust relationships and even location information [26], there is no complete protection without some form of centralized control [4]. Moreover, using fake identities may not be considered strictly as an attack; indeed it may be a rightful way of preserving privacy.

Instead of enforcing cooperation, our approach is motivated by the observation that –despite what classical game theory (cGT) predicts– cooperation *does* emerge in human societies and nature. Humans are myopic in foreseeing the outcomes of their actions and generally lack complete information about the strategic situation. Yet, they manage to achieve good results by following the best strategy locally. Evolutionary game theory (eGT) [24] argues that local adaptation of strategies, called *network reciprocity*, can help cooperation prevail in the network. With respect to ad hoc networks, even in a fully cooperative network without any Sybil attacks, the wireless medium with its inherent packet loss and collisions makes it impossible to obtain an accurate view on the intentions of the neighboring nodes, and rules out the use of cGT.

When adopting eGT for ad hoc networks, devices may employ various cooperation methods depending on their properties or the preferences of their owners, and take decisions locally based on imperfect knowledge. The outcome at the network scale will be the formation of isolated islands of cooperative devices surrounded by free riders abstaining from

relaying packets for others. The cooperative devices could follow a kind of direct/indirect reciprocity or simply cooperate without any expectation, whereas the free riders defect and try to circumvent punishments. This leads to an interesting phenomenon: *the gain of defection and cooperation strategies varies locally in real networks*. Among intelligent neighbors that can detect defective behavior, a node gains more by practicing cooperation. Still, a node better defect when surrounded by defectors (only costs, no gain) or if its neighboring nodes are unconditional cooperators (maximum gain, no costs).

Taking actions based on observed "spatial" trends prevents exploitation and, as eGT claims, promotes cooperation. It also offers an angle on ruling out Sybil attacks as one need not observe individuals (and punish them when behaving selfishly) as long as the collective "mood" can be sensed in the local neighborhood. To this end, we ask: "how can we devise meta strategies that (i) are able to adapt to locally-best pure strategy, (ii) are resistant to Sybil attacks, and (iii) do not employ individual punishments?" We investigated several meta strategies from multi-agent systems research and modified the *Stochastic Imitate Best Strategy (SIBS)* and *Win Stay Loose Shift (WSLS)* meta strategies to satisfy the three requirements given in our motivation question. As the name meta strategy indicates, nodes sporadically observe the fitness of themselves and their neighbors and then decide on which pure strategy they will employ in the subsequent round. These meta strategies had to be changed for a better fit in ad hoc networks. In the modified WSLS strategy, a node takes decisions based only on the number of packets that are relayed through it. However, nodes employing SIBS have to overhear the packet traffic to assess the fitnesses of their neighbors and choose the fittest strategy. Since it is easy to spoof the source and destination addresses inside a packet, we propose a two-hop overhearing method in which a packet is counted as relayed only if a node overhears the packet from its origin and also from its forwarder. We demonstrate the effectiveness of our two-hop overhearing policy in simulations and on a real test-bed.

The contributions of this paper can be summarized as follows:

- We modify and adapt two meta strategies, SIBS and WSLS for use in wireless ad hoc networks (Section III). We validate the proposed meta strategies as resistant to Sybil attacks (Section VI) and compare them to Tit-For-Tat (TFT) and Generous-TFT (GTFT) [23].
- We design a two-hop overhearing method for resilience to fake identities used in Sybil attacks and measure its effectiveness with both simulations and test-bed experiments (Section IV).
- We thoroughly evaluate SIBS and WSLS in their ability to adapt their environment in comparison to pure strategies. In particular, we detail experiments where nodes switch from defection to cooperation and vice versa when the neighboring nodes switch behavior. In the case of WSLS, we show that this strategy can exploit its neighbors when they are unconditional cooperators (Section V).

II. Problem Definition and Preliminaries

This section provides a brief summary of the standard model and meta strategies for the forwarder's dilemma in multi-agent systems. In particular, we examine known meta strategies according to their dependence on perfect knowledge and correct identity, and identify two suitable meta strategies for wireless ad hoc networks (SIBS and WSLS). We close the section by defining our assumptions and the problem definition.

Standard model. In the forwarder's game, a node can either cooperate –by forwarding packets of its neighbors– or defect. Consider the case of a node u. We will use set $\mathcal{V}_u$ to refer to u's neighbors, set $\mathcal{C}_u$ for cooperator neighbors, and set $\mathcal{D}_u$ for defector neighbors, hence, $\mathcal{C}_u \cup \mathcal{D}_u = \mathcal{V}_u$ and $\mathcal{C}_u \cap \mathcal{D}_u = \emptyset$. If node u decides to cooperate then its fitness will be

$$f_u = b\,|\mathcal{C}_u| - c\,|\mathcal{V}_u|, \tag{1}$$

where b is the benefit obtained from each cooperative neighbor (forwarding u's packets), and c is the cost of u's altruistic behavior payed for every neighbor ($b, c \in \mathbb{R}^+$). On the other hand, a defector does not pay any cost and its fitness (payoff) is simply

$$g_u = b\,|\mathcal{C}_u|. \tag{2}$$

Network reciprocity and meta strategies. Notice that according to (1), (2) and classical game theory defectors have always better fitnesses than cooperators, hence the defection is the best strategy. However, in the absence of complete information, nodes discover the best strategy locally. Evolutionary game theory investigates local decisions and their consequences. In a completely connected graph, defectors still have better fitnesses. However, when the graph structure does not allow random interactions among the nodes, cooperators may have better fitnesses. The evolution and dynamics of the network reveal that the probability of having a cooperator node may become higher for cooperators than defectors. This phenomenon is called as *network reciprocity* [21]. In *network reciprocity*, nodes use "*meta strategies*" to determine whether they should cooperate or defect based on the fitness in their local neighborhood. Many such meta strategies have been studied and compared in the literature, such as *imitate best neighbor, stochastic imitate best neighbor, imitate best strategy, stochastic imitate best strategy*, and *win-stay-loose-shift* [11].

Contrary to the standard setting, in wireless networks nodes do not have complete knowledge of the strategies and the fitnesses of their neighbors. Firstly, wireless communication suffers from uncertainties such as collisions and temporal link variability. Secondly, we are interested in ad hoc networks, in which the identities of the nodes are neither known nor necessarily true. This motivates us to discard any meta strategy that requires complete knowledge or identities. For instance, the *imitate best neighbor* meta strategy only works if the fittest neighbor can be identified; it inherently relies on correct identities and is therefore not considered.

Legacy SIBS and WSLS. We found two meta strategies that partially satisfy our requirements: (i) WSLS is a reactive

strategy and depends on trial and error. If a node *wins* the game, it keeps playing the same strategy. If the node *looses*, it switches strategy. It is noteworthy that, in WSLS, the network can neither be fully cooperative nor fully defective. In a fully defective network, each node infers that it is not winning against its neighbors and switches to cooperation. When a few nodes switch together, they help each other and consequently their fitnesses become better than full defection. Since a node benefits more by breaking symmetry, the network evolves in a time-varying and stochastic way. (ii) SIBS is another reactive strategy. In SIBS, each node u decides whether to cooperate or defect by imitating the aggregate neighborhood behavior with a probability computed locally as

$$P_u^{(card)} = \begin{cases} \frac{F_u^{(card)}}{F_u^{(card)}+G_u^{(card)}} & , \ F_u^{(card)} > 0 \\ 0 & , \ F_u^{(card)} \leq 0 \end{cases} \quad (3)$$

where

$$F_u^{(card)} = \sum_{v \in \mathcal{C}_u} f_u \quad \text{and} \quad G_u^{(card)} = \sum_{v \in \mathcal{D}_u} g_u$$

are the aggregate fitnesses of cooperator and defector neighbors, respectively. The superscript *card* emphasizes the use of the cardinality of the set of neighbors (which in turn implies knowledge of their identities). By analyzing SIBS in different graph families, such as lattices, small-world networks and random graphs, Ohtsuki et al. showed that the invasion of cooperators is possible when $b/c > |\mathcal{V}|$, i.e. when the benefit to cost ratio exceeds the neighborhood cardinality [21]. We have shown that this result also holds for graph families representative of ad hoc networks, such as random geometric graphs [5].

Assumptions and constraints. In this work, we assume that the network is not used for critical applications, which require reliable and secure connections. We assume that there are no malicious defector nodes, only free riders whose intention is to be regarded as cooperators to raise the chance that genuine nodes are lured into cooperation. Being malicious would deceive the purpose for the free riders as a successful break down would result in none of their packets being routed through the network.

Summary and problem definition. Sybil attacks are easy to deploy and hard to counter. When a node is in doubt, the easiest choice for it is to persist in a pure strategy, either cooperation or defection in each round. However, the heterogeneity of the network leads to islands of cooperative nodes that are spatially separated from one another. In such a topology, the payoffs of the pure strategies may vary. Therefore, local adaptation is required. The main problem we address is whether SIBS and WSLS can be modified for wireless networks to choose the locally fittest strategy. Moreover, while adapting to their environment, the modified SIBS and WSLS should not be deceived by free-riders who devise Sybil attacks.

III. Sybil-Attack Resistant Meta Strategies

In this section we argue that the original definition of fitness is *not suitable* for ad hoc networks. In wireless networks, it is not the number for cooperator and defector neighbors that matters, but the number of successfully transmitted packets. Therefore, we modify SIBS and WSLS to employ packet traffic as the fitness measure. We also show that identities become less significant as a side-effect of using packet traffic.

Unlike (1) and (2), in wireless networks benefits and costs may be expressed in terms of packet traffic; e.g., the energy consumed per packet transmission. We modify the fitness computation of meta strategies and define the fitness of node u (either defector or cooperator) as

$$f_u = g_u = b\,|\mathcal{I}_u| - c\,|\mathcal{R}_u|, \quad (4)$$

where $\mathcal{I}_u$ and $\mathcal{R}_u$ are the sets of injected and relayed packets by node u. Defectors relay none or few packets with respect to cooperators. A source node creates a data packet and transmits it to a neighbor. If the neighbor forwards the packet then we state that the source node has *injected* and the forwarder node (neighbor) has *relayed* a packet. A node gets benefit b for each *injected* packet and pays cost c for each *relayed* packet. The computed fitness value f_u does not require the knowledge of the strategy deployed by the individual neighbors. Node u observes the actions of its neighbors by overhearing their packets in two-hops and from that computes the fitness of each neighbor. This two-hop overhearing approach is further described in Section IV.

A. Stochastic Imitate Best Strategy

The main idea behind SIBS is "to go with the flow": a SIBS node periodically identifies and imitates the best strategy in its neighborhood. The node first collects samples of overheard packets in a period of time and then determines its next strategy as the one with highest fitness. As shown in Algorithm 1, nodes are classified as defectors or cooperators according to the ratio $|\mathcal{I}_u|/|\mathcal{R}_u|$ for overheard packets. The probability $P_u^{(pkt)}$ that node u cooperates is then defined as

$$P_u^{(pkt)} = \begin{cases} \frac{F_u^{(pkt)}}{F_u^{(pkt)}+G_u^{(pkt)}} & \text{if } F_u^{(pkt)} > 0 \\ 0 & \text{if } F_u^{(pkt)} \leq 0 \end{cases} \quad (5)$$

where

$$F_u^{(pkt)} = \sum_{v \in \mathcal{C}_u} f_v \quad \text{and} \quad G_u^{(pkt)} = \sum_{v \in \mathcal{D}_u} g_v\,, \quad (6)$$

and f_u is defined in (4). Note that here, sets $\mathcal{C}_u$ and $\mathcal{D}_u$ include the node u itself, respectively. The superscript "*pkt*" highlights the use of packet traffic in the formula.

The crucial operation against Sybil attacks is how $P^{(pkt)}$, $F_u^{(pkt)}$ and $G_u^{(pkt)}$ are computed. The sum operation in (6) and as demonstrated in Algorithm 1 at line 6 and line 9, limits the effect of changing identity. The algorithm shows that cooperators and defectors are grouped separately and instead of the cardinality of nodes, the fitnesses (in terms of *injected* and *relayed* packet counts) are kept in the groups. As a result, packet traffic becomes the significant factor rather than the cardinality of cooperators or defectors. Suppose that node $v \in \mathcal{V}_u$ is a defector which, for some small period, changes its identity to w and cooperates. All the packet traffic of nodes v and w will be added to $G_u^{(pkt)}$ and $F_u^{(pkt)}$, respectively. Since in truth v is a defector who does not intend to stay in cooperative state, any benefit gained by changing its identity is small. For more influence on $P_u^{(pkt)}$, node v may stay in cooperative state longer (as w). However, node v is indeed a defector,

Algorithm 1: Stochastic imitate best strategy.

Data: b: The benefit acquired by injecting a packet. c: The cost of relaying a packet. THRESHOLD: user defined parameter.
Result: The forwarding strategy of node u.

```
/* Compute fitness of cooperation and defection */
1  F_u^(pkt) = G_u^(pkt) = 0
2  foreach v ∈ V_u do
3      f_v = b|I_v| − c|R_v|
4      if |I_v|/|R_v| < THRESHOLD then          // Cooperator
5
6          F_u^(pkt) += f_v
7      else                                     // Defector
8
9          G_u^(pkt) += f_v
/* Imitate fittest strategy */
10 if max{0, F_u^(pkt) / (F_u^(pkt) + G_u^(pkt))} > rand() then
11     Cooperate
12 else
13     Defect
```

Algorithm 2: Win Stay Loose Shift meta strategy.

Data: injected: number of injected packets in previous round.
relayed: number of relayed packets in previous round.
b: Benefit acquired by injecting a packet.
c: Cost of relaying a packet.

```
1  f_u = b|I_u| − c|R_u|
2  switch type do
3      case StWSLS
4          if ((f_u < f'_u) AND (f'_u − f_u)/f'_u > rand()) OR
           rand() < 0.05 then
5              switchStrategy()
6      case pWSLS
7          if f_u < (0.75 · f'_u) OR rand() < 0.05 then
8              switchStrategy()
9  f'_u = f_u
```

hence a dilemma. Apart from grouping, we also need a packet counting mechanism that is robust to spoofed packet address fields. Our protection against spoofed addresses is the *two-hop overhearing* method that will be explained in Section IV.

In the SIBS algorithm, the *threshold* parameter distinguishes the cooperator and defector nodes (see line 4 in Algorithm 1). Although, we set *threshold* to 2 in the experiments, in general the threshold should be set according to the requirements of the nodes and the characteristics of the environment. With a high threshold, everyone will be classified as a cooperator to the benefit of the free riders. Using a low *threshold*, however, can easily lead to misconceptions as corrupted or dropped packets directly influence the sets of injected and relayed packets. On the other hands, free riders may want to maintain the reputation of a cooperator and carefully relay just enough packets to stay below the set threshold. Indeed, a defector is still forced to behave cooperatively to some extent [17].

Algorithm 1 exhibits a complexity of $O(|\mathcal{V}_u|)$, both in terms of space and time. More neighbors lead to bigger neighbor tables that SIBS needs to pass over. Space complexity is also linear with the number of neighbors, since we store two additional variables for each neighbor recording the *injected* and *relayed* counts, respectively.

Discussion. Instead of counting the packet traffic of each node, the aggregated traffic can also be used to observe the trend in the neighborhood. For instance, a node may decide to cooperate if the number of overall relayed packets is higher than a threshold. The investigation of this threshold is a separate research study. On the other hand, with the aggregated traffic method, a node does not necessarily choose the best strategy, as it is unclear which strategy is the fittest. Consider a defector neighborhood, in which a single cooperative node relays all the packets. If a node in this neighborhood employs the aggregated packet count as a decision metric, it can infer that the environment is cooperative as many packets are being relayed. However, choosing cooperation the node allows adjacent defectors to exploit its resources, which is clearly sub-optimal. In summary, meta strategies must have the ability of comparing all the available strategies.

B. Win Stay Loose Shift

WSLS is a simple meta strategy used in iterated prisoner's dilemma games [19]. In every iteration, if a node wins the game against an opponent, it keeps playing the same strategy. If not, it switches to the other strategy. WSLS is an innovative strategy which also involves random mutations. Sporadically, even when winning, the node switches to the other strategy. For instance, when surrounded with pure cooperators, a WSLS node can discover that defection may be more beneficial, where a SIBS node will always cooperate.

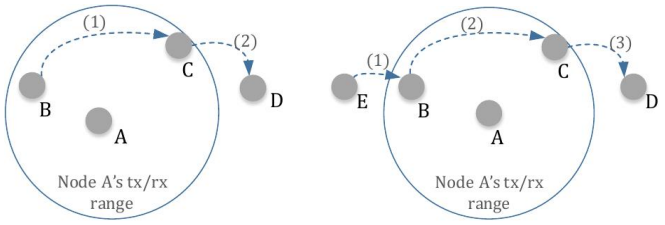
Formulating WSLS for an N-player forwarding game requires changes as the notion of winning and losing is unclear. Since we must avoid using identities, a node cannot compare itself to individual neighbors. The only option is comparing to itself. Therefore, initially a node should try both strategies and stick to the better one. Moreover, sporadically or based on changes of the current fitness, the node should again switch its strategy.

We implemented two versions of WSLS as shown in Algorithm 2. The first one: *Stochastic WSLS (StWSLS)*, at line 3, switches the strategy based on the ratio of the drop in fitness. The second one: *Plain WSLS (pWSLS)*, at line 6, limits the switch to 75% of the previous fitness value. In this work, we adopted the 75% threshold from [11]. Lastly, in both versions, the node mutates with constant 0.05 probability to explore other possibilities. Finding out the best limiting percentage and mutation rate requires an understanding of the environment and variation limits of fitnesses. Since our main contribution is not an in-depth analysis of the WSLS techniques, we omit the investigation of these parameters.

Before analyzing the performance of SIBS and WSLS, we must explain our packet overhearing method and show its performance since the robustness to Sybil attacks is inherently dependent on packet overhearing. We then pursue with an experimental analysis of the meta strategies themselves.

IV. Packet Overhearing based Fitness Estimation

In this section, we propose a fitness estimation method based on packet traffic to compute $P^{(pkt)}$ for wireless multi-hop networks. In ad hoc networks, strategy and fitness information of neighbors may not be available. Asking neighbors about their strategies and fitnesses requires trust relationships, which



(a) Node B injects a packet, and node C relays it.

(b) Both nodes, B and C relay, but only C is rewarded with a relay.

Fig. 1. Node A uses two-hop overhearing and counts packets from B to C.

strictly depend on the identity information. Therefore, a node estimates the fitnesses of itself and others, only by its own observations.

In our design, against address spoofing, nodes overhear the two-hop activity of each packet and then update the "*injected*" and "*relayed*" packet counts as defined in Section III. As depicted in Fig. 1(a), after the first transmission, node A does not take any action. Only after the second transmission from C to D, node B is attributed to have injected a packet while node C has relayed it. Additionally, assume that node C creates a packet and sends it to node D. Node A can overhear this transmission, however not the next one from node D. Consequently, the packet sent by node C is not counted as injected by node A.

The algorithm for counting injected and relayed packets is given in Algorithm 3. The algorithm is designed for the Contiki OS and is composed of two callback functions. Nodes operate in monitor mode overhearing all packets transmitted in their frequency band and channel. Address fields inside the packets are used to create the neighbor lists. The *input_sniffer* (line 1) function is called for every received packet, while the *output_sniffer* (line 12) is called for every transmitted packet. In the initial reception of a packet, it is only pushed to a set data structure. On the second reception, the relayed and injected counts for neighbors or the node itself are increased. Packets are distinguished by the use of source address, packet id and more importantly the hash of the packet for integrity check. Due to the limited memory of packets and neighbor lists, an active period of τ and maximum buffer size are introduced. For brevity, we skip the related code in the algorithm.

A. Resistance to Address Spoofing

With two-hop overhearing, we are sure that a cooperator node indeed relays a packet. Otherwise, a free rider may alter the source address of a packet and present its own injected packet as relayed. Since we overhear from both the source and the intermediate node, we conclude that the packet is relayed. Moreover, in line 8 of Algorithm 3, we determine whether initially the packet was overheard from its original source or a relay. A free rider may still alter the source address and pretend as if the packet is being relayed. However, in the algorithm, we do not give any *relayed* credits without overhearing from a source before. Other attacks by spoofing source address are as follows:

Algorithm 3: Packet overhearing algorithm for nodes in monitor mode. The algorithm is composed of two callback functions. *Input_sniffer* is called for every packet reception and *output_sniffer* for every transmission.

Data:
$storedPktList$: stores the overheard packets including the node's own packets.
sender: the previous hop address of the packet.
source: the address of the packet creator.

```
   /* INPUT SNIFFER: callback function. Called when a
      packet is overheard. */
1  def input_sniffer(sniffedPkt):
2      storedPktCopy = findPkt (sniffedPkt, storedPktList)
3      if storedPktCopy is NULL then
4          push sniffedPkt to storedPktList
5      else
6          n = findNeigh(sniffedPkt.sender)
7          n.relayed++
8          if storedPktCopy.source == storedPktCopy.sender then
               // Previously, packet was heard from
                  source.
9              n = findNeigh(storedPktCopy.source)
10             n.injected++
           // Update stored copy not to count as injected
              again.
11         storedPktCopy.sender = sniffedPkt.sender

   /* OUTPUT SNIFFER: callback function. Called when a
      packet is transmitted. */
12 def output_sniffer(sniffedPkt):
13     if sniffedPkt.source == myAddr then
14         push sniffedPkt to storedPktList
15     else
16         storedPktCopy = findPkt (sniffedPkt, storedPktList)
           /* Below is same as line 6 to 11 */
17         n = findNeigh(sniffedPkt.sender)
18         n.relayed++
19         if storedPktCopy.source == storedPktCopy.sender then
20             n = findNeigh(storedPktCopy.source)
21             n.injected++
22         storedPktCopy.sender = sniffedPkt.sender
```

(i) Spoofed relayed packet: Normally, a free rider does not relay a packet. However, if it relays and alters the source address to its own, then the free-rider node would be regarded as injecting yet another packet. Consequently, the free-rider node seems to have higher fitness and its neighbors start to become defectors as well, which is not the aim of the free rider.

(ii) Spoofed injected packet:

- **Change source to an existing cooperator node:** First of all, it is a malicious node attack, since it damages the routing protocol of the network. Secondly, if too many packets are shown to be injected from a cooperator node, then the cooperator node can be classified as defector too. Therefore, the attack should carefully be tuned. Fortunately, it is hard to tune in a distributed environment. Nodes can hardly guess which part of the traffic is being overheard by others due to the two-hop overhearing and channel conditions.
- **Change source to a defector node:** In this case, defectors would be regarded as having better fitness values, which turns the neighbors to defectors.
- **Change source to a dummy node (Invisibility attack):** A free rider alters the source address of its

packets to a dummy value. Hence, this free rider node will be considered as if it does not inject any packet and operates with zero fitness, and therefore simply becomes *invisible*. However, overall, the relayed packets by cooperator nodes still keep climbing, which leads to lesser fitness of the cooperators. Therefore, SIBS eventually switches to defection.

All the above attacks are particularly an issue for the SIBS meta strategy since WSLS only considers its own fitness and counts only the packets that pass over the node itself. Especially for the *invisibility attack*, the immunity of WSLS can be incorporated to SIBS by merging SIBS with WSLS. While determining the fittest strategy in the neighborhood, the node can give more weight to the traffic that passes over itself. We plan to investigate the hybrid strategy in the future.

Our two-hop overhearing classifies a packet when the next-hop of the packet is inside the transmission range of the node. As a consequence, some portion of the traffic may be missed. In the next sub-section, we present the performance of two-hop overhearing in terms of captured traffic.

B. Experiments on Overhearing

Two-hop overhearing works best when two neighboring nodes are close to each other and consequently share most of their neighbors. The expected ratio of the intersection of the communication coverage area of two neighboring nodes u, v assuming the unit-disk model in 2D is $E[\frac{Area(u,v)}{Area(u)}] \approx 0.59$ [14]. When v sends packets to its neighbors, u overhears with the probability of $P_{heard} = 0.59$. Besides this approximation, we study two-hop overhearing with the Cooja simulator [22], part of the Contiki OS. Periodically, a node creates a packet and sends it through the network via *random-walk* routing, in which the next hop is randomly chosen from the neighbors except the previous hop. The packet travels till it reaches the destination or as far as the hop-limit allows. See Table I for simulation parameters.

We present the ratio of overheard packets, R_{heard} in Fig. 2. Every node uses the two-hop overhearing method to count the packet traffic of its own and its neighbors. In R_{heard}, the denominator is the node's own count of its traffic and the numerator is its neighbor's perception. There are 100 nodes, each with ≈ 10 neighbors, in total ≈ 1000 data points exist for

TABLE I. SIMULATION PARAMETERS FOR TWO-HOP OVERHEARING.

Operating System	Contiki
Number of nodes	100
Area size	$50 \times 50\ m^2$
Transmission (tx) Range	$10, 15, 20\ m$
Routing	Random Walk
MAC Layer	CSMA
Hop-limit	$Random(1-4)$
Packet generation rate	$4 + Random(1-3)$ sec
Packet Size	100 bytes
injected/relayed threshold	2
Repetitions	100

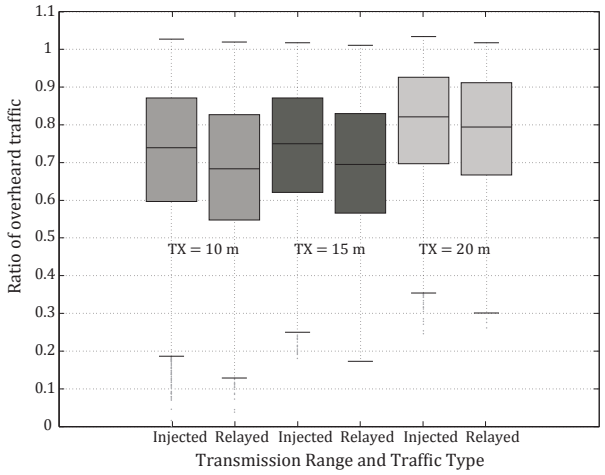


Fig. 2. Ratio of packet traffic sniffed by the two-hop overhearing method. Injected and relayed traffic for different transmission (tx) ranges are depicted.

each box-plot. In the box-plots, the central mark indicates the median and the edges of the box correspond to the 25th and 75th percentiles. R_{heard} varies based on the transmission (tx) range. Since the deployment area is fixed, longer transmission ranges result in the coverage of higher fractions of the network. Consequently, the second hops of the packets are chosen more in the neighborhood. Combined with the border effect, overall, the overhearing performance is better than the approximation; it reaches up to $60 - 70\%$ of classified packet traffic. After a packet reaches a border, it is pushed back into the network, which keeps the packet in the neighborhood of the overhearing nodes. Additionally, the box-plots also indicate a variance incurred due to the proximity difference of the nodes to each other. When nodes are closer to each other, they have more neighbors in common. Hence, more packets are overheard. In some box-plots, R_{heard} becomes greater than 1. The reason is that, sometimes the sender of a packet does not overhear the packet's next transmission due to packet collisions while neighbors can still overhear.

In Fig. 2, we present the *injected* and *relayed* traffic separately. The overhearing ratio of *relayed* packets is less than the *injected* for all the transmission ranges. In Algorithm 3, at line 8, if the initial source is the same as the sender then we increase the *injected* count. However, we do not increase the *relayed* count, since the two-hop overhearing method defines relaying only if the packet was heard previously. Therefore, there is a bias towards the *injected* count. On the other hand, after a packet is injected to the network it may travel more than one hop, each of which is counted as a relay. Fig. 1(b) presents a case where node B relays a packet, but it is not considered. Since node A cannot hear any transmission of the packet before it reaches to B, it cannot claim that B has relayed the packet. In our simulations with a hop-limit that is uniformly distributed between $1 - 4$, the average number of injected packets is *half* ($66/130$ for $tx = 10\,m$) of the relayed packets. Consequently, although not seen in the figure, the effective bias is towards *relayed* packets; more *relayed* packets are overheard than *injected* ones.

Additionally, we deployed the two-hop overhearing algo-

rithm in a real-world testbed located in our department at TUDELFT. The testbed consists of 108 SOWNet G301 nodes with CC1101 radios operating at 868 MHz. The deployment area is almost rectangular with dimensions 14.65 by 81 meters. For more detail please refer to [25]. We experimented with different transmission power levels. As explained in [25], at -34 dBm the network becomes partially connected and at all values above it, the network gets fully connected. The average ratio of injected and relayed packets, $R_{injected}$ and $R_{relayed}$, is given in Table II. The correspondence of the ratios with the simulation results depends on the transmission range. When the transmission power is low, the effect of asymmetric links is observed as a low perception of the traffic. However, with a larger transmission power (higher dBm), links become more symmetric and neighbor degree increases establishing a mesh topology of the deployed nodes. Thus, most of the packet traffic occurs in the overhearing range of the nodes.

TABLE II. AVERAGE RATIO OF OVERHEARD PACKETS IN A REAL-WORLD TESTBED WITH DIFFERENT TRANSMISSION (tx) POWERS.

tx power (dBm)	$R_{injected}$	$R_{relayed}$
-34	0.51	0.45
-15	0.59	0.51
-6	0.71	0.64
0	0.76	0.73

The two-hop overhearing method is not perfect, even the overhearing ratios of relayed and injected packets are not the same. Nevertheless, it still captures a large fraction of the packet traffic which still reveals the cooperation behavior of the neighborhood. In the next section, we demonstrate the performance of SIBS and WSLS in discovering the locally fittest strategy with two-hop overhearing.

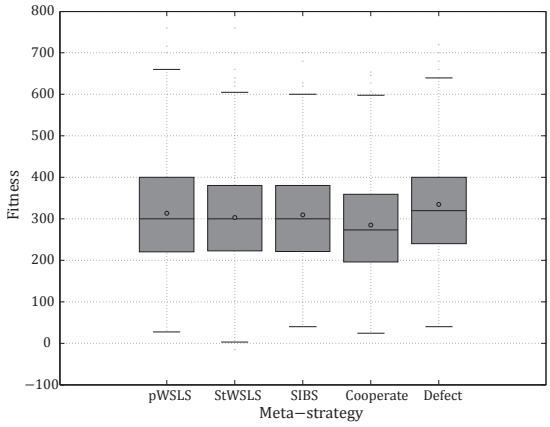
V. THE LOCAL ADAPTATION OF META STRATEGIES

Contrary to what it is customarily assumed, real networks are composed of devices with diverse intelligence and information. As a consequence, strategies become heterogeneous and spatially varying. To evaluate the adaptation of meta strategies, we assume that some nodes are more intelligent and have more information about who is a defector or cooperator. These "*ideal*" nodes employ an incentive technique like indirect or direct reciprocity among themselves and they trust each other. They are able to retaliate against defectors by selectively dropping their packets and relaying the others. Hence, it is wiser to cooperate when surrounded by *ideal* nodes.

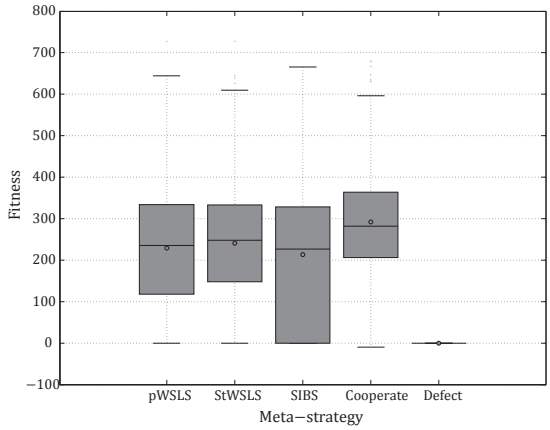
In the evaluation we observe the fitness of a node which employs different meta- and pure-strategies. In different scenarios where either defector or cooperator is the best strategy, we demonstrate how meta strategies discover the best one.

A. Experiments on Local Adaptation

To observe the adaptation of the meta strategies to the environment, we performed experiments on the Cooja simulator. We traced the behavior of a single node over 50 rounds in 20 different experiments. The simulation parameters are the same as overhearing experiments and given in Table I except, the transmission range and benefit-to-cost-ratio are set to $tx = 10\,m$ and ${}^{b}/_{c} = 20$. We set the ${}^{b}/_{c}$ ratio to fulfill the condition of ${}^{b}/_{c} > |\mathcal{V}|$ (See Section II). We have two scenarios.



(a) 50% defectors and 50% cooperators.



(b) 50% defectors and 50% ideals.

Fig. 3. Average fitness of a node when it follows different strategies.

In the first one, the network is comprised of 50% defector and 50% cooperator nodes, while in the second one, 50% defector and 50% ideal nodes are used. In each round, after the convergence of $P^{(pkt)}$ (See Section V-B), the traced node reconsiders its strategy according to the overheard packets in the previous round. Throughout the 50 rounds of a simulation, nodes choose either cooperation or defection in each round so that rounds are classified as cooperative and defective.

In the first scenario, there is no ideal node to punish defectors and hence, defection is better than cooperation. On other hand, in the second scenario cooperation is better. When we consider the ratio of rounds that the node chooses the fittest strategy in both scenarios, we observe that all meta strategies discover the fittest strategy with similar performance. Respectively, the ratio of cooperative rounds for pWSLS, StWSLS and SIBS are $0.35, 0.4, 0.32$ for the first scenario and $0.79, 0.84, 0.71$ for the second scenario. In the first, nodes tend to follow defection, while they cooperate in the second. Fig. 3 provides more information on the achieved fitness values in

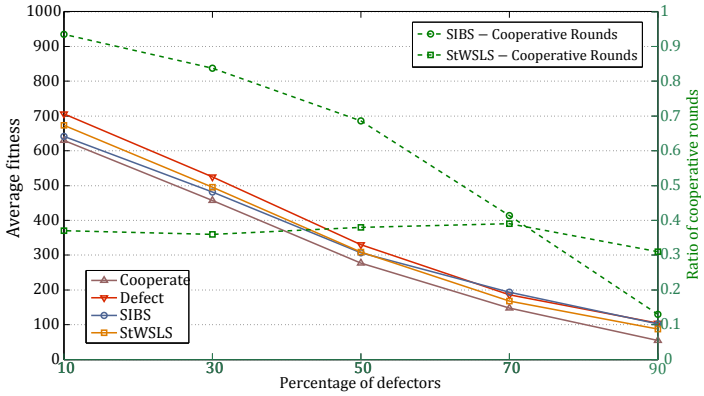


Fig. 4. Average fitnesses (left - solid lines) and ratio of cooperative rounds (right - dashed lines) of meta strategies according to the defector density.

the two scenarios. Meta strategies achieve comparable fitness values to the best strategies of each scenario. Notice that in the second scenario, *ideal* nodes isolate defectors whose fitness is zero (See Fig. 3(b)). It shows that following pure-defection does not pay off always. As a result, instead of following pure cooperation and pure defection, with meta strategies nodes are able to detect the fittest strategy in their locality and adapt to it. Lastly, we should note that pWSLS and StWSLS perform similarly. Therefore, in the rest of the paper we only report on StWSLS.

In the scenarios we deployed 50% by 50% from each type to minimize the domination of one. When we break the symmetry, SIBS effectively reveals the change while WSLS hardly realize the new structure. We re-experimented the first scenario with different defector densities (in percentage): $10, 30, 50, 70$ and 90. The ratio of cooperator rounds with increasing defector densities for SIBS and StWSLS are given in Fig. 4. Consider the right-Y axis, with high defector density, SIBS chooses cooperation lesser, while StWSLS has almost constant ratio for all deployments. The consequence of not choosing the advantageous strategy is also shown in Fig. 4 in terms of average fitness values (left-Y axis). Defection outperforms others since there is no *ideal* node in this scenario. With respect to pure cooperation both of the meta strategies offer better fitness values. When the defector percentage exceeds 50%, SIBS starts performing better by not choosing cooperation. We can see the same trend also in average fitnesses. Additionally, the decrease in overall fitness with high defector densities indicates the effect of selfish behavior on a network.

A meta strategy may also consider exploiting its neighbors when they are pure cooperators. We deployed a network composed of fully cooperative nodes, and placed one adaptive node amidst them. SIBS immediately chooses cooperation. However, with mutations, WSLS discovers that it is surrounded by pure cooperators and observes that defection has better payoff. As a result, the WSLS-enabled node chooses cooperation on average 48% of the experiments. WSLS is an innovative strategy that it is able to try other strategies that are not used in the neighborhood. However, for the same reason, WSLS also leads to instability and the network never becomes fully cooperative.

B. Convergence of $P^{(pkt)}$ estimation

Nodes overhear their surrounding neighbors to estimate $P^{(pkt)}$, but for how long? The duration should be as short as possible to conserve energy, yet long enough for accurate $P^{(pkt)}$ estimation. The convergence of estimating $P^{(pkt)}$ depends on the neighbor degree and the packet generation rate. Having no a-priori knowledge of the traffic pattern, in this paper we use a random walk routing, in which the next hop is determined randomly. According to the Coupon Collector's problem [12], to be able to send m packets to every neighbor, a node needs $k \log k + (m-1)k \log\big(\log k\big) + O(k)$ trials where k is the neighbor degree ($|\mathcal{V}|$). In our case, the average packet generation rate is 6.5 $pkts/sec$, $k \approx 10$. Considering the packet losses if we listen for two rounds, $m = 2$, on average a node should keep overhearing $6.5 \times [10 \log 10 + (1)10 \log\big(\log 10\big) + O(10)] \approx 268$ sec. In preliminary simulations, with a variable $k \approx 10$, we observed this phenomenon around 300 sec. We have to note that these results are only representative for the random walk routing. The effect of different routing policies to estimation cooperation is an open problem.

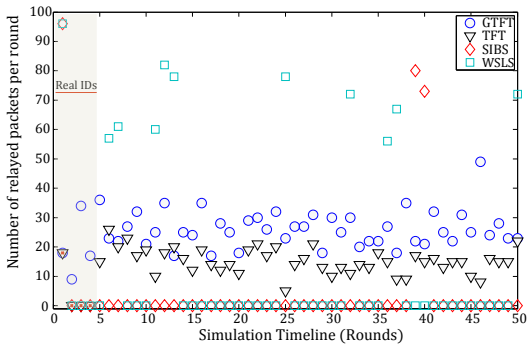
The SIBS and WSLS meta strategies are shown to be able to discover the fittest strategy in their local neighborhood. Next, we present the robustness of these meta strategies to Sybil attacks.

VI. Robustness to Sybil attacks

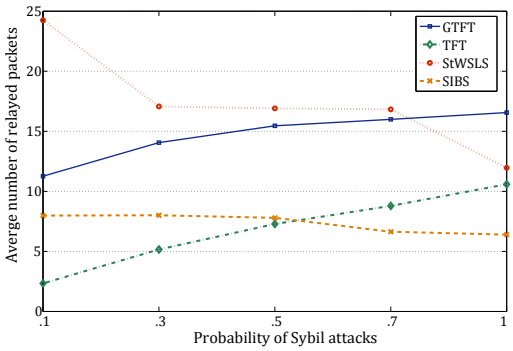
In this section, we demonstrate the response of our meta strategies to Sybil attacks involving fake identities. To give a measure of performance, we also implemented and compared our methods to two simple and pretty successful direct-reciprocity methods, the Tit-For-Tat (TFT) and Generous-Tit-For-Tat (GTFT) [23]. A node with the TFT or GTFT strategy starts relaying the packets of its neighbor and follows the neighbor's strategy as observed in the previous round. The difference between both strategies is that GTFT becomes sporadically generous and gives a new chance to defector nodes in case of defection from both sides. The amount of the relayed packets is chosen as a metric for Sybil attack robustness where less indicates better robustness.

We deployed a network composed of 80% defectors and 20% cooperators, and an adaptive node that employed GTFT, TFT, StWSLS and SIBS in different experiments. Unlike cooperators, defectors start fabricating new fake identities after the 4th round. Fig. 5(a), depicts the number of relayed packets for each strategy for one experiment. SIBS and StWSLS start with relaying all the packets and then they switch to defection. SIBS is more stable and keeps defecting, while StWSLS sporadically tries cooperation. On the other hand, although GTFT and TFT selectively relay packets, they keep relaying in every round. In the first 4 rounds, TFT as highlighted with dots, manages to punish all the defectors. However, as soon as defectors start fabricating fake identities, TFT starts relaying again. The lack of individual punishments in WSLS and SIBS leads to bursts of relayed packets. When they are cooperative, they relay more packets than GTFT and TFT. However, WSLS and SIBS are able to handle fake identities and immediately recognize defection as the fittest strategy.

In a second set of experiments, we present the average number of relayed packets while defectors initiate attacks with



(a) Number of relayed packets per round. Defectors constantly fabricate identities.



(b) Average number of relayed packets vs. ratio of Sybil attacks.

Fig. 5. Relayed packets of meta strategies and direct-reciprocity methods. The network is composed of 80% defectors and 20% cooperators.

a probability (See Fig. 5(b)). Sporadically, defectors decide to change their identity with the given probability. We should note that when nodes keep changing their identities, many packets are sent without a valid destination. Hence, high Sybil attack probabilities effectively decrease the number of relayed packets. We see this effect clearly in StWSLS and SIBS. On the other hand, although there exists an overall decrease in relayed packets, TFT and GTFT still incur increased numbers of relayed packets.

VII. Related Work

The mechanisms that explain the survival cooperators in nature are [20]: *Kin Selection*, *Direct Reciprocity*, *Indirect Reciprocity*, *Network Reciprocity* and *Group Selection*. Among these, *Direct Reciprocity* and *Indirect Reciprocity* are famous in the wireless networks research. However, in this work we concentrate on *Network Reciprocity*. Spatial structure of the networks combined with "social viscosity" enables the spread of cooperation.

Direct reciprocity depends on the repeated interaction of nodes and follows the *"You scratch my back and I'll scratch yours"* principle. However, in a public wireless network, random encounters may not suggest a future interaction [6]. Additionally, it is obvious that identity is strictly required. In order to punish identity changes, as a bootstrapping phase, every node should "pay their dues" before injecting packets. However, a node may need immediate operation or it may employ fake identities to protect its privacy. On the other hand, our meta strategies tolerates the fake identities and does not require a bootstrapping phase.

Indirect reciprocity makes use of reputations based on the principle of *"I scratch your back and someone else will scratch mine"*. Contrary to direct reciprocity, indirect reciprocity can prevail even in the case of random encounters. However, distribution of reputation in a decentralized system is quite challenging. There are an abundance of different proposals like CORE [18], SORI [10] and others [13], [15]. Indirect reciprocity requires true identity information. However, even so, slandering attacks like good/bad mouthing and whitewashing are hard to avoid. Free-rider nodes retaliate against other nodes that report the node as defector by announcing those nodes as defectors too. By using a good mouthing attack, free riders cooperate among each other and announce good reputation about each other. In our work, we do not depend on reports from neighbors and hence, our meta strategies are not susceptible to slandering attacks. Moreover, accumulating reputation and distributing it takes time and leads to energy consumption in a bootstrapping phase [7], whereas our scheme does not need extra message exchange for any kind of information.

Additionally, there is also **credit exchange**, which depends on the secure transfer of any type of currency in return for forwarding a packet. Nuglet [2] is a budget counting mechanism. When a node helps others by forwarding their packets, its budget increases and later the budget is used to inject packet to the network. Similarly, there are many other works [1], [16] that achieves cooperation by employing credit-exchange. However, ad hoc networks require special tamper-resistant security modules to count the currencies or centralized entities with powerful encryption have to be deployed.

Even with the existence of central control, authentication and identification of the constrained devices is still challenging [3]. IETF-CoRE working group aims to provide a secure network for Class-1 and above devices. Class-0 devices, which have less than 10KB of RAM, cannot employ security protocols. However, it is hard to claim that employing demanding cryptographic algorithms like asymmetric keys are impossible. While some old works [8] require special hardware, recent works [9] deploy simple asymmetric elliptic-curve cryptography just by software on MSP430X low-power processors. As we have demonstrated, our proposed meta strategies operate together with such *ideal* nodes that have enough cognitive capabilities to authenticate identities.

VIII. Conclusions

Without cooperation multi-hop communication is not possible. Unfortunately, the forwarder's game proves that if the network is composed of selfish nodes, defection prevails throughout the network. Many research studies have been carried out to incentivize cooperation by rewards and punishments. However, these works do not consider the heterogeneity

of networks, which stems from diverse cognitive limits of nodes and the methods they employ to build trust. Moreover, ambiguity in the wireless medium and hard to prevent Sybil attacks exacerbate the exploit of rewards and punishments. In this work, we modify and adapt two meta strategies, SIBS and WSLS, from multi-agent systems so that they do not depend on individual rewards and punishments. Nodes follow pure cooperation or pure defection in rounds. They determine which one to follow by considering the fitnesses of their neighbors and themselves. Contrary to literature, fitness computation is based on the observed packet traffic, which we measure by using a novel two-hop overhearing method. A side-effect of using packet traffic is observed as resilience to Sybil attacks.

Our experiments demonstrate that SIBS and WSLS are able to adopt the locally fittest strategy. Armed with two-hop overhearing, they are resistant to Sybil and other identity related attacks. With respect to SIBS, WSLS is rather easy to implement, more responsive to attacks and innovative in terms of switching to defection in a cooperative neighborhood. WSLS requires less traffic to be traced and consequently consumes less energy. However, on average, SIBS handles the Sybil attacks better than the WSLS. When the percentage of defectors increases SIBS offers better average fitnesses.

As a future work, the invasion of cooperation in networks should also be re-considered for wireless networks. Multi-agent systems research argues that a fully cooperative network is possible with SIBS when the $^{b}/_{c} > |\mathcal{V}|$ condition holds where $|\mathcal{V}|$ is the neighbor degree, b is the benefit from an altruistic act, while c is the cost of it. We need to re-validate or revise this condition since contrary to literature that is based on neighbor cardinality, our fitness measure is based on packet traffic. A second future work is on the effect of routing. In this work we simplified routing to concentrate on adaptation. However, with different routing schemes especially two-hop overhearing method is expected to behave differently.

ACKNOWLEDGMENTS

This work was supported by the Trans-sector Research Academy for complex Networks and Services (TRANS) project. Andreas Loukas was supported by the Dutch Technology Foundation STW and the Technology Program of the Ministry of Economic Affairs, Agriculture and Innovation (D2S2 project). Dr. Onur has been partially supported by METU BAP-08-11-2014-025.

REFERENCES

[1] L. Anderegg and S. Eidenbenz, "Ad hoc-vcg: A truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents," in *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '03. New York, NY, USA: ACM, 2003, pp. 245–259.

[2] L. Buttyán and J.-P. Hubaux, "Stimulating Cooperation in Self-Organizing Mobile Ad Hoc," *Mobile Networks and Applications*, vol. 8, no. 5, pp. 579–592, October 2003.

[3] X. Chen, K. Makki, K. Yen, and N. Pissinou, "Sensor network security: a survey," *IEEE Communications Surveys Tutorials*, vol. 11, no. 2, pp. 52–73, Second 2009.

[4] J. Douceur, "The sybil attack," in *Peer-to-Peer Systems*, ser. Lecture Notes in Computer Science, P. Druschel, F. Kaashoek, and A. Rowstron, Eds. Springer Berlin Heidelberg, 2002, vol. 2429, pp. 251–260.

[5] Y. Durmus and E. Onur, "Imitation as the simplest strategy for cooperation," in *Personal Indoor and Mobile Radio Communications (PIMRC), 2012 IEEE 23rd International Symposium on*, Sept 2012, pp. 863–869.

[6] M. Felegyhazi, J.-P. Hubaux, and L. Buttyan, "Nash equilibria of packet forwarding strategies in wireless ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 5, pp. 463–476, May 2006.

[7] S. Ganeriwal, L. K. Balzano, and M. B. Srivastava, "Reputation-based framework for high integrity sensor networks," *ACM Transactions on Sensor Networks*, vol. 4, no. 3, pp. 15:1–15:37, Jun. 2008. [Online]. Available: http://doi.acm.org/10.1145/1362542.1362546

[8] G. Gaubatz, J.-P. Kaps, and B. Sunar, "Public key cryptography in sensor networksrevisited," in *Security in Ad-hoc and Sensor Networks*, ser. Lecture Notes in Computer Science, C. Castelluccia, H. Hartenstein, C. Paar, and D. Westhoff, Eds. Springer Berlin Heidelberg, 2005, vol. 3313, pp. 2–18.

[9] C. Gouvła, L. Oliveira, and J. Lpez, "Efficient software implementation of public-key cryptography on sensor networks using the msp430x microcontroller," *Journal of Cryptographic Engineering*, vol. 2, no. 1, pp. 19–29, 2012.

[10] Q. He, D. Wu, and P. Khosla, "Sori: a secure and objective reputation-based incentive scheme for ad-hoc networks," in *Wireless Communications and Networking Conference, 2004. WCNC. 2004 IEEE*, vol. 2, 2004, pp. 825–830.

[11] L.-M. Hofmann, N. Chakraborty, and K. Sycara, "The evolution of cooperation in self-interested agent societies: A critical study," in *The 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, ser. AAMAS '11, Richland, SC, 2011, pp. 685–692.

[12] R. Isaac, *The pleasures of probability*. Springer, 1995.

[13] J. J. Jaramillo and R. Srikant, "A game theory based reputation mechanism to incentivize cooperation in wireless ad hoc networks," *Ad Hoc Networks*, vol. 8, no. 4, pp. 416 – 429, 2010.

[14] I. Khalil, S. Bagchi, and N. B. Shroff, "Liteworp: Detection and isolation of the wormhole attack in static multihop wireless networks," *Computer networks*, vol. 51, no. 13, pp. 3750–3772, 2007.

[15] Y. Liu, K. Li, Y. Jin, Y. Zhang, and W. Qu, "A novel reputation computation model based on subjective logic for mobile ad hoc networks," *Future Generation Computer Systems*, vol. 27, no. 5, pp. 547 – 554, 2011.

[16] M. Mahmoud and X. Shen, "Pis: A practical incentive system for multihop wireless networks," *Vehicular Technology, IEEE Transactions on*, vol. 59, no. 8, pp. 4012 –4025, October 2010.

[17] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '00. New York, NY, USA: ACM, 2000, pp. 255–265.

[18] P. Michiardi and R. Molva, "Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks," in *Proceedings of the IFIP TC6/TC11*, Deventer, The Netherlands, 2002, pp. 107–121.

[19] M. Nowak, K. Sigmund *et al.*, "A strategy of win-stay, lose-shift that outperforms tit-for-tat in the prisoner's dilemma game," *Nature*, vol. 364, no. 6432, pp. 56–58, 1993.

[20] M. A. Nowak, "Five rules for the evolution of cooperation," *Science*, vol. 314, no. 5805, pp. 1560–1563, 2006.

[21] H. Ohtsuki, C. Hauert, E. Lieberman, and M. a. Nowak, "A simple rule for the evolution of cooperation on graphs and social networks." *Nature*, vol. 441, no. 7092, pp. 502–5, May 2006.

[22] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-level sensor network simulation with cooja," in *IEEE Conference on Local Computer Networks*, Nov 2006, pp. 641–648.

[23] V. Srinivasan, P. Nuggehalli, C. Chiasserini, and R. Rao, "Cooperation in wireless ad hoc networks," in *Proc. of the INFOCOM 2003.*, vol. 2, march-april 2003, pp. 808 – 817.

[24] G. Szab and G. Fth, "Evolutionary games on graphs," *Physics Reports*, vol. 446, no. 46, pp. 97 – 216, 2007.

[25] M. Woehrle, M. Bor, and K. Langendoen, "868 MHz: a noiseless environment, but no free lunch for protocol design," in *9th int. conf. on Networked Sensing Systems*, ser. INSS, jun 2012, pp. 1–8.

[26] Y. Yu, K. Li, W. Zhou, and P. Li, "Trust mechanisms in wireless sensor networks: Attack analysis and countermeasures," *Journal of Network and Computer Applications*, vol. 35, no. 3, pp. 867 – 880, 2012.