A Framework for Packet-Droppers Mitigation in OLSR Wireless Community Networks

Francesco Saverio Proto, Andrea Detti, Claudio Pisa, Giuseppe Bianchi
University of Rome Tor Vergata
{firstname.lastname}@uniroma2.it

Abstract—**Wireless community networks are mesh networks built by volunteers which own, configure, and manage their wireless node. Such networks are prone to either unintentional (e.g. misconfiguration) or intentional node misbehavior. This paper proposes a fully distributed trust-based routing framework, tightly integrated with OLSR, which is the most exploited routing protocol in real world wireless community networks. The framework, designed to be modular for easy upgrade, relies on active probes, hidden in the normal data traffic through adaptation of steganography techniques. The combination of path-wide measurements into a distributed trust-framework, preliminarily based upon the well known EigenTrust mechanism, permit to infer whether, and which, packet-droppers (i.e. nodes misbehaving at the data plane) affect the network forwarding operation. The resulting per-node trust values are then transformed into suitable "weights" provided as input to the OLSR protocol for mitigation through re-routing. A simulation-based performance evaluation shows that the proposed framework appears already effective in detecting and circumventing packet-droppers, despite the relative simplicity of the preliminarily considered algorithms.**

I. Introduction

A wireless community network is a self-provisioned mesh network composed of fixed wireless nodes. The distinguishing characteristic of a wireless community network is the impossibility to rely on a centralized network management and monitoring framework, as almost each node composing the mesh is owned by a distinct entity, frequently an individual person. The large majority of wireless community deployments, including but not limiting to [1], [2], [3], [4], rely on the OLSR [5] routing protocol. A very active public domain developers' community is tailoring the OLSR operation for the specific needs of wireless mesh networks and for the very cheap employed devices.

Nowadays, there exist wireless community networks of more than 10.000 nodes [4]. Even if relatively small with respect to commercially operated networks, these scales have become extremely challenging especially for management purposes. As the responsibility of controlling each node is ultimately up to its owner, misconfiguration impairing the network behavior may frequently emerge. Now, if routing problems are already non trivial to detect (e.g. taking advantage of the network-wise state each node maintains using information broadcasted by the OLSR protocol), data plane misbehavior is even more challenging, especially when it does not reflect into a routing anomaly. This is for instance the case of a node firewall misconfiguration: the node may still properly forward one-hop traffic, such as the OLSR control traffic, but may prevent the forwarding of multi-hop data traffic. In this case the routing is safe but the data delivery is unintentionally harmed.

On top of this, we should further consider that *intentional* node misbehavior cannot be ruled out in a relatively large community network, where it has become practically impossible for any participant to know (and trust) every other node. Intentional attacks to the OLSR protocol have been extensively addressed in both literature and in community deployments, and consensus has been reached on means for securing the OLSR routing plane [6], [7], [8]. Conversely, a consensus solution appears still lagging for what concerns disruption of data forwarding, where even in the case of a safe routing plane, a misbehaving node might agree to forward data packets but might fail to do so (we name such a node as "packet-dropper").

A. Design requirements

We believe that approaches devised to detect and mitigate data delivery misbehavior in wireless community networks should be designed with in mind the following requirements.

First, the widespread adoption of OLSR calls for mechanisms that are readily integrated in the OLSR operation, i.e. the level of node trustworthiness computed by the trust-mechanism should drive the weighting of the links of the OLSR topology. Since OLSR forwarding occurs hop-by-hop on the basis of the routing table that each node has autonomously computed, it is necessary that all nodes practically operate on the same weighted topology to avoid routing loops in the network. This implies that the trust mechanism must provide *a global vision of the level of nodes' trustworthiness*.

Second, since the IEEE 802.11 WLAN is the widespread radio technology used in the deployment of wireless community networks, the trust mechanism must be *fully compliant with IEEE 802.11*.

Third, in order to limit radio interference and improve the network throughput, there is an increasing adoption of nodes with multiple radio cards and directional antennas, therefore the trust mechanism must be *compliant with multiple radio configurations*.

Fourth, the approaches should be the least invasive as possible and yield *minimal computational load* to be easily supported by cheap devices.

B. Our contribution

This paper brings the following contribution:

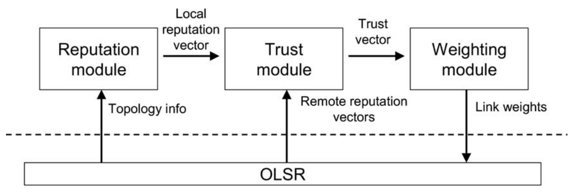

Fig. 1. Trust-based routing framework

- the proposal of a modular framework that interworks with the OLSR routing protocol and ensures a reliable data delivery against the presence of packet-droppers;
- the design of a mechanism that uses ordinary packets as "hidden" probes, and enables nodes to evaluate the reputation of other nodes only by exploiting its own data traffic and without the need of "overhearing" other node's traffic [9];
- the design of a link weighting function that integrates trust in the OLSR metric and practically allows to enforce routing on the path with the best path-trustworthiness, where path-trustworthiness is the trustworthiness of the *worst* node of a path;
- the preliminary assessment of the performance of the proposed approaches via NS2 simulation in the presence of UDP traffic (the extension to TCP traffic requires to further address more targeted attacks such as malicious SYN packets dropping, and is left to future work).

II. Proposed OLSR-based framework

The proposed framework (Fig. 1) is formed by OLSR and three modules, named: reputation-module, trust-module and weighting-module. The reputation-module and trust-module form the trust-mechanism that assesses the nodes' trustworthiness, while the weighting module properly maps the level of nodes' trustworthiness to the weight of links of the OLSR topology.

The reputation-module of a node S computes the level of reputation that node S has of all other nodes. A node S evaluates the reputation of a node D by monitoring the correctness of data forwarding by node D. Reputation values computed by S are collected in a reputation-vector that is flooded in the network and locally sent to the trust-module.

The trust-module collects local and remote reputation-vectors and by combining them it computes a trust-vector. The i-th element of the trust-vector represents the level of trustworthiness of the i-th network node. The trust-vector is sent to the weighting-module.

Finally, the weighting-module makes use of the values of the trust-vector elements to derive and to enforce the weights of the links of the OLSR topology.

A. Reputation-module

The reputation-module assesses the reputation that a node S has of other nodes. A widespread solution to this issue consists in overhearing the traffic directed to neighboring nodes in order to evaluate the correctness of their forwarding operation [9], [10], [11], [12], [13]. Although effective in several scenarios, overhearing may be difficult in the presence of multiple directional antennas or when multi-rate transmissions are allowed, and these cases are common in a wireless community network scenario.

To support cases when overhearing is difficult we devise a reputation-module by adopting a *path-wide probe-based* approach that best fits connectionless UDP traffic. The underlying idea is the following: a node S selects another node D and sends to D probe packets. When the destination node D receives a probe packet it sends back a probe response [1].

An important issue consists in how to deploy probe packets. We promote an approach based on "implicit" probes, hidden in normal traffic, and whose details are presented in Section III. This prevents a malicious attacker to operate by correctly forwarding probe packets, while dropping data packets without losing reputation. Note that probe responses do not require protection (i.e., they can be made explicit) as an attacker dropping responses would reduce its own reputation level. Moreover, the use of normal traffic at the endpoints does not require overhearing and limits computational complexity, since each node performs computation only on its own data traffic rather than on all forwarded traffic. It is worth to observe that in the presence of deployments where the goal is not to defend against malicious attackers, but only monitor the emergence of unintentional forwarding misbehavior, explicit probes may be further deployed to increase the number of paths controlled, and hence the effectiveness of the reputation framework.

When the probe response is received, node S increases the reputation of *all* nodes of the path S-to-D; otherwise, if the probe response is not received within a time-out (e.g., 500ms), the reputation of all nodes of the path S-to-D is decreased[2]. Which are the nodes of the S-to-D path is inferred by the OLSR routing module .

In practice, the reputation r_{si} that a node S has of another node I is equal to the probability to get a response for a probing packet sent by S and that is deemed to pass through node I. We evaluate this probability through a moving average, based for instance on the last 10 samples. If a probe succeeds, then the value of the sample is 1; otherwise sample value is 0. In addition, a soft-state mechanism resets r_{si} to the default value, i.e. 1, if no probing (i.e., no traffic) interesting the node I has been done since a valuable amount of time, for instance 60 seconds.

The reputation values r_{si} are collected in a *reputation-vector*. This vector is flooded into the OLSR network by using *reputation-messages*, a new type of OLSR message defined by us, which contains the reputation-vector in its payload and is signed by the originator S. Flooding is achieved by using the OLSR default forwarding algorithm [5].

We observe that it is fair to increase the reputation of all the nodes of the path in case of a successful probing, because

[1]Note that in case overhearing was feasible, our path-wide probe-based implementation would work with or would be replaced by an overhearing approach

[2]In case of non-symmetric paths (e.g., in presence of OLSR ETX extension) also the reputation of the nodes of the reverse path has to be increased or decreased

indeed all nodes succeed in data forwarding. Conversely, it may be unfair to decrease the reputation of all the nodes of the path in case of failure of the probing, because indeed only a single node may misbehave. This means that some node may obtain a reputation that is worse that the deserved one. Such an unfair decrease of reputation is clearly an adverse side effect of path-wide probing. However, the reputation-worsening is limited by combining locally and globally the results of different probes. Local mitigation occurs when node S exchanges probes with different destinations; in doing so a "good" node obtains a decrease of its reputation when the probe is on a path that includes both the good and a misbehaving node, but the good node obtains an increase in its reputation every time the misbehaving node does not belong to the probe path. Furthermore, reputation-worsening is globally limited because, as we explain below, the trust-module combines the reputation values estimated by other nodes.

B. *Trust-module*

The aim of the trust-module is to combine the reputation-vectors provided by the local and remote reputation-modules in order to compute a global level of trustworthiness for each network node. We base our implementation of the trust-module on the well-know EigenTrust framework [14]. The EigenTrust framework fits well to our scenario, as departing from a formalization of the (natural) concept of transitive trust, leads to values of node trustworthiness that are global over the network, i.e. each network node computes the same values[3].

For sake of completeness we only report the formulas of the basic EigenTrust algorithm we used. First of all normalized reputation values c_{si} are defined as follows:

$$c_{si} = \frac{\max(r_{si}, 0)}{\sum_i \max(r_{si}, 0)} \tag{1}$$

By defining t_i as the trustworthiness of node i, $\vec{t} = [t_i]$ as the *trust-vector*, $\vec{e}$ as a vector representing uniform probability distribution over all nodes, $C = [c_{ji}]$ as the matrix containing all the normalized values of the reputation-vectors and δ as a small value (e.g., 0.001), then the EigenTrust algorithm computes $\vec{t}$ through the iteration reported in algorithm 1.

C. *Weighting-module*

The weighting-module aims at weighting the links of the OLSR topology on the basis of the values contained in the trust-vector $\vec{t}$. We devise an implementation of the weighting-module with the following goal: *a longer path without malicious nodes is always preferred to a shorter one with a malicious node*. We point out that such a goal requires a careful design of a trust-to-weight mapping function, since the (OLSR) shortest-path algorithm evaluates the cost of a path by *summing* the level of trustworthiness (i.e., link weights) of belonging nodes and we wish this cost to resemble the level of trustworthiness of the *worst* nodes.

[3]It is worth to observe that in [14] the authors specify also a reputation system to compute r_{si} that unfortunately does not fit well in case of packet-droppers

Algorithm 1 Basic EigenTrust Algorithm

```
$\vec{t}_0 = \vec{e}$
**repeat**
    $\vec{t}_{k+1} = C^T \vec{t}_k$
**until** $abs(\vec{t}_{k+1} - \vec{t}_k < \delta)$
$\vec{t} = \vec{t}_{k+1}$
```

We propose a trust-to-weight mapping composed by two steps: in the first step the trust values t_i are quantized in three symbolic values of trustworthiness, named tq_{high}, tq_{medium}, tq_{low}; in the second step link weights are derived by these symbolic levels of trustworthiness.

The symbolic quantized value tq_i of a trust value t_i is reported hereafter:

$$tq_i = \begin{cases} tq_{high} & \text{if } t_i \geq \frac{1}{N} \\ tq_{medium} & \text{if } \frac{1}{2N} \leq t_i < \frac{1}{N} \\ tq_{low} & \text{if } t_i < \frac{1}{2N} \end{cases} \tag{2}$$

where N is the total number of nodes in the network. The reason behind the selection of the threshold values of eq. 2 is that if all the N nodes of the network are well-behaving, then their normalized trust is $\frac{1}{N}$. By increasing the number of misbehaving nodes, the trust of good nodes increases beyond $\frac{1}{N}$ and the trust of misbehaving nodes tends to zero. We use three quantization levels to address the case of partial packet droppers, that will fall in the tq_{medium} level.

Now we insert the quantized values of node trustworthiness in the OLSR metric. We define the quantized trustworthiness tq_{si} of a unidirectional link between node S and node I as the lower quantized trustworthiness among tq_s and tq_i. The weight w_{si} of the unidirectional link S-to-I is a function of tq_{si} and therefore we have three possible numerical weights, named w_{low}, w_{medium}, w_{high}, that are respectively assigned in case of tq_{si} is equal to tq_{low}, tq_{medium} or tq_{high}. The values of w_{low}, w_{medium}, w_{high} should ensure that, whatever is the path stretch in terms of network hops, shortest-path routing selects the most trusted path, where the path-trustworthiness is equal to the (quantized) trustworthiness of the worst node of the path. To achieve this goal it is enough that the values w_{low}, w_{medium}, w_{high} comply with the following two conditions:

$$\begin{cases} ML \cdot w_{high} < w_{medium} \\ ML \cdot w_{medium} < w_{low} \end{cases} \tag{3}$$

where ML is the maximum path length in the network. For instance, by choosing ML=10 the possible values would be: w_{low}=100, w_{medium}=1, w_{high}=0.01.

Finally, we observe that albeit we have considered only three levels of quantization (low, medium, and high) the reasoning that we followed can be repeated for more quantization levels.

III. Hiding probes

To hide probe packets we leverage steganographic techniques for creating *implicit* probes "hidden" within the plain data traffic. To achieve this task, it is required that each sender and destination pair shares a dedicated secret. Secret sharing may exploit an eventual PKI deployed for securing the routing plane [6] or may be done through dedicated asymmetric cryptographic schemes[4].

Let us assume that S sends UDP traffic to a destination D, and that S and D share a dedicated secret k_{SD}. When an UDP/IP packet P is sent from S to D, it is recognized as probe by both source and destination nodes (and only by themselves, because of the secrecy of k_{SD}) if the condition $HMAC_{k_{SD}}(P) \leq threshold$ holds. Here, HMAC is a keyed hash construction based on a secure one-way hash function, keyed with the node pair secret, and threshold is a configuration value that permits to configure the percentage of probes (1/32 in our specific case, by setting threshold equal to 1/32 of the maximum HMAC value). Furthermore, we recall that the HMAC must be applied to the content of the packet P which is not mutable during its forwarding in the network (for instance, its TTL which must be hence excluded from the hash computation).

Since a malicious packet-dropper does not know k_{SD}, it cannot compute the HMAC and hence detect whether the packet is a probe or not. The only possible weakness occurs when packets P which satisfy the condition of being probes happen to appear duplicated in the data stream, as in this case a node keeping track of all the delivered packets and recognizing them as probes based on the occurrence of a probe response would be able to identify the ones subsequent to the first one. Note that the first packet remains protected, as this is disclosed to be a probe only too late from the point of view of a malicious node. This issue could be in principle avoided by adding freshness to *every* packet payload - e.g. in the form of a random trailer. However this would need to remove such extra data at destination and complicate the implementation. In practice, we argue that the presence of application layer headers which contain changing information (such as sequence numbers and timestamps in RTP/UDP streams) is sufficient to avoid this concern.

The probe response is a special packet constructed by the destination. It does not need to be hidden, as the lack of probe response delivery to the source would imply node misbehavior (and hence a node dropping probe responses would harm its own reputation). Therefore, it is constructed as an explicit UDP message that contains a (different) keyed message authentication code for the original packet P. As such, the source is able to link the probe response to the packet P initially send, whereas a malicious node is not able, missing the secret, to forge fake probes. Any duplicated or invalid probe response is discarded by S.

Finally, we observe that an implicit probing mechanism does not fully address the scenario of TCP traffic. Indeed, a malicious packet-dropper might immediately discard packets that establish a connection thus preventing the exchange of traffic and probes. Moreover, also considering initial TCP SYN packets as probes would be ineffective because a malicious node could forward these packets but drop the remaining traffic, thus blocking further probing. We argue that a possible approach for devising a reputation-module for TCP traffic could be based on the analysis of connection level phenomena such as TCP timeout.

[4]For instance, using Diffie-Hellmann (but the same holds for many other schemes, e.g. identity-based cryptography, bilinear pairings, etc), each node can preliminary notify to all the remaining nodes (e.g. through broadcast delivery) its public Diffie-Hellman parameter, so that each remaining node can asynchronously compute a per-node-pair shared secret.

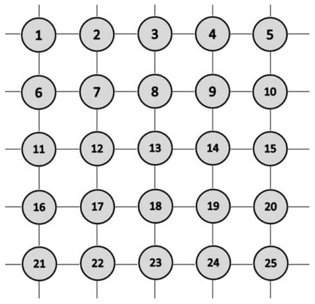

Fig. 2. Simulation scenario

IV. Performance evaluation

We test our algorithm in a mesh network formed by 25 fixed nodes disposed as in figure 2. We use Network Simulator 2 (v2.34), enriched by the OLSR modules of [15]. A node can directly communicate only with its horizontal or vertical neighbors. MAC layer is IEEE 802.11 operating at 11 Mbps. A simulation lasts 300 sec; beginning at 20 sec and every 10 sec each node sets up a CBR/UDP session with a random destination; UDP packets have a length of 1452 bytes and the UDP bit-rate is 220 kbps. Regarding the probing performed by the reputation-module, we consider a threshold equal to 1/32 of the HMAC maximum value, i.e. in average a probe every 32 delivered packets. Therefore during an UDP session of 10 seconds we have on average 6 probes; the reset of a local reputation value occurs after 60 seconds of probing inactivity. Regarding the model of the malicious node we assume that it drops all UDP packets and advertises a bad reputation for all other nodes.

Figure 3 reports the time evolution of the quantized trust value tq_i of nodes 12, 13 and 14, in case node 13 drops all the UDP packets to be forwarded while the other network nodes correctly forward packets. Since the beginning of UDP traffic, i.e. 20 sec, the malicious node 13 gets a low level of trust, while neighbor nodes 12 and 14 get a high trust level. We also observe that good nodes 12 and 14 get during a very small period of time a wrong reputation and this is an evidence of the worsening effects coming out from the path-wide probing approach that we followed (see section II-A); conversely sometimes nodes 13 gets a high level of trust due to the temporary reset of the soft-state mechanism.

Figure 4 shows the number of packets delivered to every node for forwarding. When trust-based-routing is disabled (upper plot) the three dropper nodes intercept a significant number of packets and the total amount of forwarded packets in the network is lower. Enabling trust-based-routing the droppers are identified and skipped, leading to a higher number of forwarded packets for the other nodes of the network.

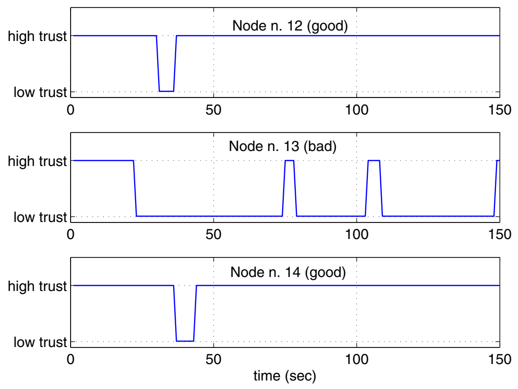

Fig. 3. Time evolution of quantized trust values in case node 13 is a packet-dropper

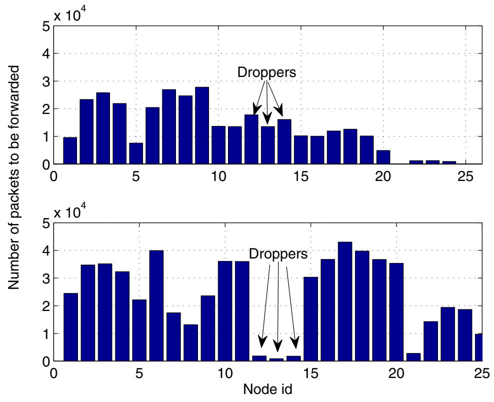

Fig. 4. Number of packets to be forwarded by network nodes when nodes 12, 13 and 14 are packet-droppers, in cases of absence (upper plot) and presence (lower plot) of trust-based-routing

Figure 5 shows the effectiveness of the trust-based-routing in coping with many independent packet-droppers[5]. We measure the effectiveness E as the reduction of packets intercepted by droppers when the trust-based-routing is activated, in formula:

$$E = 1 - \frac{P_{trust}}{P_{plain}}$$

where P_{trust} and P_{plain} are the number of packets intercepted by packet-droppers in case of trust-based-routing and in case of plain OLSR routing respectively. In case of a single packet-dropper, we observe a reduction of the number of intercepted packet that is about 97%. We do not obtain a reduction of the 100% since the reset timeout of reputation-modules allows traffic to be temporary intercepted by the droppers. Obviously, by increasing the number of droppers we have a decrease of E that however remains beyond 90% up to 4 packet-droppers.

Finally, figure 6 shows the impact of the reputation reset timeout and of the number of probes per second in case of 3 packet-droppers. We observe that both parameters have a marginal impact on system performance on the condition that a too short timeout (e.g., lower than 20 sec) and too few probes per second (e.g., lower than 3 probes) are avoided.

[5]The ids of the dropping nodes are: 1-dropper: {13}, 2-droppers: {12,14}, 3-droppers: {12,13,14}, 4-droppers: {7,9,17,19}, 5-droppers: {7,9,13,17,19}, 6-droppers: {7,8,9,17,18,19}, 7-droppers: {3,7,8,9,17,18,19}

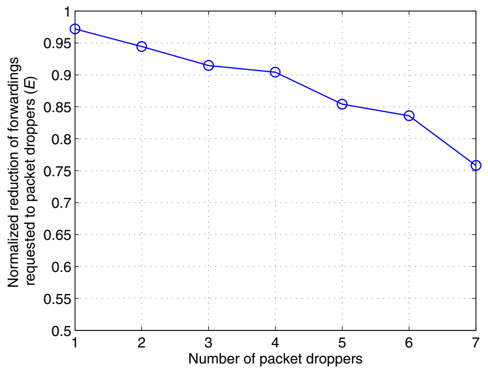

Fig. 5. Normalized reduction (E) of number of packets to be forwarded by packet-droppers in case of trust-based-routing with respect to the case of plain OLSR, versus the number of packet-droppers

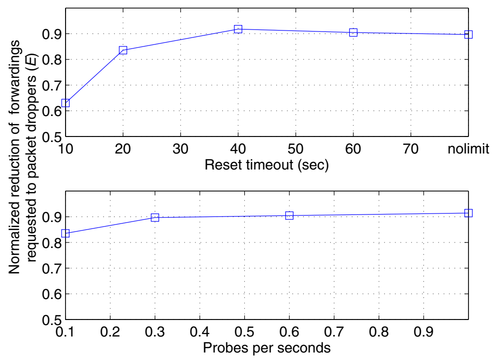

Fig. 6. Normalized reduction (E) of number of packets to be forwarded by 3 packet-droppers in case of trust-based-routing with respect to the case of plain OLSR, versus the duration of the reset time-out and the frequency of probe packets

V. Related work

This section surveys design aspects of relevant and recent work in literature aimed at mitigating attacks on the data plane in wireless multi-hop networks.

In [9] the authors introduce the idea of two modules called *watchdog* to identify misbehaving nodes, and *pathrater* to help the routing protocol (DSR), in choosing routes that avoid these nodes. To identify misbehaviors, the watchdog “overhears” neighbors, by putting its network interface in promiscuous mode, to check if data packets are actually forwarded by them, and then this information is used by the path rater as a metric to compute the best routes. In other works similar neighbor monitoring mechanisms have been exploited, together with protocols devised to spread reputation values over the network: CONFIDANT [10] makes use of alarm messages, while CORE [11] propagates this information through a *provider and requestor* scheme. Moreover, in [12], the monitoring is enhanced by using unkeyed hashes in the DSR route discovery phase.

In [16] the proposed scheme copes with attackers that independently fail to forward data packets or are not honest in propagating trustworthiness information to the other nodes. A hop by hop acknowledgement scheme, based on overhearing, is used to gather first-hand trust information in the assumption that the underlying routing protocol is based on source routing. When insufficient information is available the node can query a set of recommender nodes to obtain second-hand information. The amount of second-hand information used can be tuned to achieve either the goal of accuracy or speed of detection.

The usage of the watchdog mechanism is instead limited, in [17], to the monitoring of broadcast packets, while for unicast packets two-hop cryptographic acknowledgments are employed. The proposed scenario is a DSR MANET, where a PKI is assumed to be in place, and both attacks to the data and control planes are considered. The gathered reputation information is then used as the basis for an accusation-based collaborative node isolation mechanism.

In [18] authors introduce the concept of witnesses, i.e. nodes that do not belong to the data forwarding path but that are able to monitor (by overhearing) nodes belonging to it. Witnesses send reports to the packet source on the forwarding behavior of the monitored nodes that belong to the data path. The authors provide an analytical model and study its performance. In the special case of a network topology that does not allow for witnesses, the mechanism falls back to the watchdog approach.

In [19], the authors devise a reputation-based mechanism for AODV MANETs with very mobile and sparse nodes based on EigenTrust. Each node collects information on other nodes checking, with a mechanism similar to the previously mentioned watchdog, if data is actually forwarded. This reputation information, spread over the network, together with information on the centrality of the nodes, i.e. how wide is their view of the network, is then used to classify nodes into zones, after applying a variant of the EigenTrust algorithm, to compute trust values.

The key difference between our work and the current state of the art, is that the proposed framework enables nodes to evaluate the reputation of other nodes without the need of overhearing, this makes possible to use the framework in case of multiple directional antennas setups or when multi-rate transmissions are used. Moreover, the proposed path-wide probe-based approach requires the use of a link state routing protocol.

VI. Conclusion

This paper proposes a modular trust-based routing framework for OLSR based Wireless Community Networks. The proposed approach does not require overhearing, and limits computational load since a node only performs packet-level trust computation upon packets of its traffic. Such virtues are achieved by assessing the nodes' reputation (in terms of proper forwarding of data packets) via a path-wide approach leveraging "hidden" active probes deployed into the normal data packet. A preliminary performance evaluation shows that the proposed approach is effective, even if it is based on a very simple reputation algorithm (Section II-A) and it is targeted to detect "packet droppers". As future work, we plan to improve the extent and effectiveness of the framework in correctly and rapidly identify misbehaving nodes, by devising more sophisticated inference algorithms capable to better assess the forwarding operation of an intermediate network node using only end-to-end measurements (e.g. using methodologies similar to those exploited by network tomography research [20]).

VII. Acknowledgements

This work has been partially supported in the frame of the European Project FP7-ICT FLAVIA, contract number 257263.

References

[1] "Freifunk: non commercial open initiative to support free radio networks in the German region," http://www.freifunk.net/.

[2] "Funkfeuer free net," http://www.funkfeuer.at/.

[3] "Ninux.org wireless community network," http://ninux.org/.

[4] "Guifi.net," http://guifi.net/en.

[5] T. Clausen and P. Jacquet, "Optimized link state routing protocol (OLSR)," IETF RFC 3626, October 2003.

[6] C. Adjih, D. Raffo, and P. Mühlethaler, "Attacks against OLSR: Distributed key management for security," in *2005 OLSR Interop and Workshop*, Ecole Polytechnique, Palaiseau, France, July 28–29 2005.

[7] C. Adjih, T. Clausen, P. Jacquet, A. Laouiti, P. Mühlethaler, and D. Raffo, "Securing the OLSR protocol," in *Proceedings of the 2nd IFIP Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net 2003)*, Mahdia, Tunisia, June 25–27 2003.

[8] D. Raffo, C. Adjih, T. Clausen, and P. Mühlethaler, "An advanced signature system for OLSR," in *Proceedings of the 2004 ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '04)*. Washington, DC, USA: ACM Press, October 25 2004, pp. 10–16.

[9] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM, 2000, pp. 255–265.

[10] S. Buchegger and J.-Y. Le Boudec, "Performance analysis of the CONFIDANT protocol," in *MobiHoc '02: Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*. New York, NY, USA: ACM, 2002, pp. 226–236.

[11] 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 Sixth Joint Working Conference on Communications and Multimedia Security*. Deventer, The Netherlands, The Netherlands: Kluwer, B.V., 2002, pp. 107–121.

[12] K. Paul and D. Westhoff, "Context aware detection of selfish nodes in DSR based ad-hoc networks," in *proceedings of IEEE GLOBECOM*, 2002, pp. 178–182.

[13] S. Buchegger, C. Tissieres, and J. Le Boudec, "A test-bed for misbehavior detection in mobile ad-hoc networks - how much can watchdogs really do?" in *Proceedings of IEEE WMCSA 2004*, English Lake District, UK, December 2004.

[14] S. D. Kamvar, M. T. Schlosser, and H. G. Molina, "The EigenTrust algorithm for reputation management in P2P networks," in *WWW '03: Proceedings of the 12th international conference on World Wide Web*. New York, NY, USA: ACM, 2003, pp. 640–651.

[15] W. Cordeiro, "Olsr modules for ns2 (olsr-md, olsr-etx, olsr-ml, olsr)." [Online]. Available: http://www.inf.ufrgs.br/~wlccordeiro/resources/olsr/

[16] C. Zouridaki, B. L. Mark, M. Hejmo, and R. K. Thomas, "E-hermes: A robust cooperative trust establishment scheme for mobile ad hoc networks," *Ad Hoc Networks*, vol. 7, no. 6, pp. 1156–1168, August 2009.

[17] D. Djenouri and N. Badache, "On eliminating packet droppers in manet: A modular solution," *Ad Hoc Networks*, vol. 7, no. 6, pp. 1243–1258, August 2009.

[18] S. Yang, S. Vasudevan, and J. Kurose, "Witness-based detection of forwarding misbehaviors in wireless networks," in *Wireless Mesh Networks (WIMESH 2010), 2010 Fifth IEEE Workshop on*. IEEE, June 2010, pp. 1–6.

[19] S. Zakhary and M. Radenkovic, "Reputation-based security protocol for MANETs in highly mobile disconnection-prone environments," in *Proceedings of WONS 2010, Kranjska Gora, Slovenia*, February 2010.

[20] N. Duffield, F. L. Presti, V. Paxson, and D. Towsley, "Inferring link loss using striped unicast probes," in *IEEE INFOCOM 2001*. IEEE, April 2001, pp. 915–923.