Dynamic Tunnel Routing for Reliable and Resilient Data Forwarding in Wireless Sensor Networks

Andrea Munari†, Wolfgang Schott†, and Yee Wei Law*
† IBM Research GmbH - Zurich Research Laboratory (ZRL), Zurich, Switzerland
* Dept. of E&E Engineering, The University of Melbourne, Parkville, Victoria 3052, Australia
email: {una, sct}@zurich.ibm.com, yee.wei.law@gmail.com

***Abstract*—In this paper, we propose a novel dynamic routing strategy called *tunnel routing* that enables efficient and resilient data forwarding from sensors to a destination in wireless sensor networks. Instead of establishing a single static path, tunnel routing identifies a set of trusted sensor nodes that form a *tunnel* between the source and the destination. Terminals in the tunnel cooperate with each other to reliably route data and to defend against compromised nodes that may maliciously drop or re-direct packets. To achieve this, not only do sensor terminals forward data, but also they act as *support nodes*, providing on-demand retransmissions in order to combat channel fading, and initializing multi-path data forwarding towards the destination in case they detect a misbehavior of a forwarding node. Simulation results demonstrate the advantages of the proposed scheme.**

I. Introduction

Wireless Sensor Networks (WSNs) have the potential to enable a wide range of new applications in both the military and civilian domain, such as building and battlefield surveillance, environmental monitoring, disaster prevention, and homeland security. These networks usually consist of a large number of sensor nodes that are limited in processing capability and battery power. Since the nodes are often scattered over a wide geographic area and deployed in a harsh and even hostile environment, reliable and resilient data forwarding techniques have to be applied to securely route the sensed data hop-by-hop from a remote sensor to the destination. To this aim, two main strategies have been considered so far, which we call static and dynamic routing. In *static* routing, a route discovery procedure is exploited to identify a single path from a source to a destination, which is later used to forward data packets. With this approach, as soon as one of the nodes along the discovered path becomes unavailable due to a weak radio link, battery depletion, or security attacks, a new route construction procedure is needed to establish an alternative connection. On the other hand, *dynamic* routing does not rely on the establishment of a single path, but rather chooses a forwarding node among a set of potential candidates depending on a routing metric. Up to now, dynamic routing algorithms have mostly been proposed for networks where each node knows its own geographic location (geographic routing). In this case, a node that has data to forward simply broadcasts a request packet with the geographic coordinates of the destination to its neighbors. The receivers of this message exploit their topological knowledge to calculate the advancement that they offer towards the sink and contend among each other to elect the next hop closest to the destination. This routing strategy strongly enhances the robustness against node unavailability, yet the need for geographic information limits its applicability.

This paper describes work that has been partly supported by the Australian Research Council Research Network on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP).

In this paper, we present a novel dynamic routing strategy called *tunnel routing* that enables reliable and resilient data forwarding in WSNs without requiring geographic location information. Tunnel routing exploits a route construction procedure to identify a family of trusted sensor nodes that form a *tunnel* between a source and a destination and that later contend among each other to act as data forwarders. Moreover, terminals within the tunnel may act as *support nodes* by providing on-demand re-transmissions of data packets to combat channel impairments (cooperative relaying), or by initiating multi-path data forwarding towards the destination in case they detect a misbehavior caused by a selective forwarding attack. The proposed protocol offers the robustness of dynamic routing without requiring any additional effort in terms of location management. Moreover, our solution is resilient to packet dropping and selective forwarding attacks and degrades gracefully in the presence of attacks to several nodes.

The remainder of the paper is organized as follows: Section II discusses related work followed by the network assumptions in Section III. Section IV describes details of the proposed dynamic tunnel routing protocol. Section V provides the simulation results followed by our conclusions in Section VI.

II. Related Work

In this Section, we present some examples of static and dynamic routing, highlighting their key features as well as stressing their limits. Moreover, we briefly describe their resiliency to outsider attacks and selective forwarding insider attacks.

A. Static Routing

Ad hoc On demand Distance Vector (AODV) is one of the best known routing protocols for wireless ad-hoc and sensor networks [1]. With AODV, a route is established only when a node has data to deliver to a destination $\boldsymbol{D}$ and a path to it is not already known. In this case, the source $\boldsymbol{S}$ broadcasts a Route REQuest (RREQ) packet containing the address of the sink. Each node that receives this message checks its routing table to determine whether it can provide a path to $\boldsymbol{D}$. If not, the RREQ is rebroadcast. Otherwise, the terminal, potentially $\boldsymbol{D}$, initiates a return procedure by sending a unicast Route REPly (RREP) packet to the node it has received the RREQ message from. Upon receiving a RREP, a node updates its routing table, as it becomes aware of whom it has to forward packets addressed to $\boldsymbol{D}$ to, and iterates the procedure. Once the RREP message reaches $\boldsymbol{S}$, the path is built and can be used to deliver data. AODV does not need any geographic information to work properly, since the destination is located by flooding messages through the network. On the other hand, the algorithm provides a single path to $\boldsymbol{D}$. Therefore, as soon as one of the nodes in the identified route is not available anymore due to harsh radio channel conditions, security attacks or sleep mode induced activities, the whole path collapses. We remark that these issues are not specific to AODV, but they are also valid for all single- or multi-path protocols that see a route as

a predetermined sequence of forwarding nodes [2], [3], and may lead to a severe performance degradation in WSNs. Static routing approaches can be protected against outsider security attacks by employing the Localized Encryption and Authentication Protocol (LEAP) [4]. This protocol has been designed to enable in-network processing, while restricting the impact of compromised nodes to their neighborhood. LEAP supports the establishment of several types of symmetric encryption keys on each sensor node, among them a cluster key and a pair-wise link key. A cluster key is shared among neighbor nodes and is applied in AODV to encrypt the control messages exchanged during route establishment. On the other hand, during data forwarding a pair-wise link key protects information exchange between the nodes. While per-link encryption provides excellent protection against outsider attacks, the risk of breaking the path to the destination by compromising a single node with a selective forwarding attack is high.

To defend against insider security attacks such as selective forwarding [5], the use of static multiple-path routing protocols has been proposed. These schemes establish several paths to a destination in order to provide redundant and thus more reliable data forwarding. For example, the algorithm proposed in [6] ensures that most of the data packets are forwarded through the most efficient paths to the destination, while a small fraction of the packets are spread out to increase the chance that at least one of them reaches the destination. However, robustness against data forwarding attacks comes at the expense of an increased energy consumption, and thus a reduced lifetime for WSNs.

B. Dynamic Routing

Geographic Random Forwarding (GeRaF) [7] is an interesting example of a dynamic routing algorithm for WSNs where nodes are aware of their geographical location. According to this scheme, a source that has data to deliver to a destination $\boldsymbol{D}$ transmits a Request-To-Send (RTS) packet with its and $\boldsymbol{D}$'s geographical coordinates. Terminals that decode this message evaluate the geographic advancement they offer towards $\boldsymbol{D}$, compute a metric that is inversely proportional to this value, and set a backoff timer proportional to the metric. The node closest to the destination replies first with a Clear-To-Send (CTS) message, proposing itself as data forwarder. On the contrary, all other terminals overhear the CTS and give up the contention. Once the procedure has been accomplished, the source sends the data packet to the identified forwarding node, which then iterates the algorithm. Another example of dynamic geographic routing tailored for WSNs is RoCoDiLe [8]. As in GeRaF, nodes that receive a data forwarding request contend among each other to become the next-hop. However, RoCoDiLe employs a modified metric that considers three factors: geographic advancement towards the destination, radio channel conditions, and remaining battery energy at the node. While the first term follows the usual geographic routing approach, the second one prevents terminals that are experiencing a poor channel from becoming next-hop. Finally, the last term ensures that nodes running out of energy do not actively participate in the routing process unless strictly necessary, so as to maximize the average sensor lifetime. RoCoDiLe offers important advantages over simple geographic routing schemes [9], also thanks to the exploitation of cooperative relaying [10] and leapfrogging techniques.

Strategies such as GeRaF and RoCoDiLe do not separate path construction and data forwarding. Instead, these protocols route data packets by locally selecting the best-suited next forwarding node according to a metric, without knowing in advance the complete route that will be followed to reach the destination. The capability of dynamically choosing the next forwarding node among a set of potential candidates instead of relying on a static path strengthens these approaches with respect to node unavailability due to weak radio link, battery depletion, or insider security attacks. Nevertheless, the discussed solutions achieve this robustness by assuming geographical knowledge at sensor nodes, which limits their applicability. Dynamic routing protocols can also be protected against outsider security attacks by using LEAP. However, in case of using RoCoDiLe, the cluster key shared among neighbor nodes has to be used not only during the exchange of control messages but also when transmitting data packets. Since per-link encryption cannot be provided, the data protection against outsider attacks is worse as in case of using static routing or GeRaF. Dynamic routing strategies are more robust against selective-forwarding insider attacks because weak radio links and compromised nodes can easily be bypassed.

III. Network assumptions and Thread model

In our work, we consider WSNs composed by a large number of sensor nodes that are scattered over a wide geographic area. In this context, data generated at a source may have to undergo several hops in order to reach the intended destination. Nodes that can be reached in one or two hops from a sensor $\boldsymbol{X}$ are referred to as the 1- or 2-hop neighbors of $\boldsymbol{X}$. We assume that the packet transmission between any two sensor nodes is modeled by the input-output relation: $y(t) = h(t)a(t) + z(t)$, where $y(t)$ and $a(t)$ are the complex baseband receive and transmit signal; $h(t)$ models the radio channel and takes into account path loss and fading effects, and the additive noise $z(t)$ is a zero-mean white complex Gaussian process. We apply LEAP at the initial deployment of each sensor node to establish symmetric shared keys between terminals for providing confidentiality and authentication in the WSN. In particular, this key management protocol discovers all 1-hop neighbors of the newly deployed node, and establishes on it one pair-wise link key for secure point-to-point communication to each 1-hop neighbor and one cluster key that is shared with all its 1-hop neighbors. We assume that each node encrypts its generated control messages and data packets with the cluster key in order to allow 1-hop neighbor nodes to overhear the information transmission. In addition, we assume that each node possesses not only a list of all trusted 1-hop neighbors, but also a verified list of all 2-hop neighbors. The latter list is obtained by executing an extended LEAP algorithm [11]. Several insider attacks can be launched to compromise one or several nodes of the WSN [5]. Our protocol is designed so that it can reliably defend against forwarding insider attacks, where malicious sensor nodes may refuse to forward certain data packets or simply drop all of them ensuring that they are not forwarded any further. The proposed tunnel routing scheme can also cope with blind-letter attacks if a list of all trusted 2-hop neighbors is available at the sensor node.

IV. Tunnel Routing

Tunnel routing is a novel dynamic routing strategy for reliably and resiliently forwarding data from a source $\boldsymbol{S}$ to a destination $\boldsymbol{D}$ in a multi-hop WSN. The algorithm first performs a tunnel discovery procedure in order to locate the destination in the network, identify a set of potential data forwarders, and gather information later required to route data towards $\boldsymbol{D}$. Each node stores such information in an internal routing table, whose entries comprise the address src_{addr} of the node that initiated the tunnel discovery procedure, the destination address $\text{dest}_{\text{addr}}$, a sequence number seq_{ID} that uniquely identifies the tunnel discovery request, the hop distances $\text{n}_{\text{hop,S}}$ and $\text{n}_{\text{hop,S}}$ of the node from $\boldsymbol{S}$ and $\boldsymbol{D}$, and a parameter TimeStamp that defines the tunnel lifetime. Once

the tunnel has been constructed, data packets can be routed from $\boldsymbol{S}$ through to $\boldsymbol{D}$ by applying a dynamic data forwarding scheme. In this Section, we first describe the tunnel discovery procedure. Then, we discuss the dynamic data forwarding scheme and finally we present functionalities that can be added to the protocol to cope with errors caused by the radio channel and selective-forwarding attacks.

A. Tunnel Discovery Procedure

When a node $\boldsymbol{S}$ has data to deliver to $\boldsymbol{D}$ and an active tunnel to $\boldsymbol{D}$ has not yet been established, a new entry in its routing table is created, with fields src_{addr}= S, $\text{dest}_{\text{addr}}$= D, $\text{n}_{\text{hop,S}}$= 0, $\text{n}_{\text{hop,D}}$= ∞, and TimeStamp = 0. The field seq_{ID} is initialized to seq_old + 1, if an entry for $\boldsymbol{D}$ with seq_{ID} = seq_old already exists, or to 1, otherwise. Once the entry has been written, $\boldsymbol{S}$ broadcasts an encrypted Route REQuest (RREQ) message to all its 1-hop neighbors, comprising the parameters source address, destination address, sequence number seq = seq_{ID} , and hop count number n_{hop} = $\text{n}_{\text{hop,S}}$. A node that decodes this packet, compares the received parameter n_{hop} to a threshold $\text{n}_{\text{hop,max}}$. If $\text{n}_{\text{hop}} \geq \text{n}_{\text{hop,max}}$, the message is discarded to avoid long and unreliable paths during data forwarding. Otherwise, the node checks its routing table. If an entry for the node pair $\boldsymbol{S}$-$\boldsymbol{D}$ already exists with the same or a smaller sequence number than the one received, the message is discarded. Otherwise, the node creates a routing table entry with fields src_{addr}= S, $\text{dest}_{\text{addr}}$= D, seq_{ID} = seq, $\text{n}_{\text{hop,S}}$= n_{hop}+1, $\text{n}_{\text{hop,D}}$= ∞, and TimeStamp = 0. Any terminal that has the $\text{n}_{\text{hop,S}}$field set to a non-zero value for a given node pair $\boldsymbol{S}$-$\boldsymbol{D}$ is said to have the *Forward Flag* switched on for that couple. Once the table entry has been stored, the node creates an encrypted RREQ message with an updated n_{hop} parameter and transmits it. The described procedure is iterated until $\boldsymbol{D}$ is reached. When the destination receives the RREQ packet, it creates a new entry in its routing table with fields src_{addr}= S, $\text{dest}_{\text{addr}}$= D, $\text{n}_{\text{hop,S}}$= n_{hop}, $\text{n}_{\text{hop,D}}$= 0, seq_{ID} = seq, TimeStamp = CurrentTime. Then, $\boldsymbol{D}$ broadcasts an encrypted Route REPly (RREP) message to all its 1-hop neighbors, comprising the parameters src_{addr}= S, $\text{dest}_{\text{addr}}$= D, seq = seq_{ID} , n_{hop} = $\text{n}_{\text{hop,D}}$, and Time = TimeStamp. Neighbors that decode this message perform the test on $\text{n}_{\text{hop,max}}$ and, if the requirement is fulfilled, check if they have switched on the *Forward Flag* for the pair $\boldsymbol{S}$-$\boldsymbol{D}$. If so, they further determine if the sequence number stored in the seq_{ID} field equals the one provided by the RREP message. If the tests fail, the node discards the received message. Otherwise, the $\text{n}_{\text{hop,D}}$and TimeStamp fields of the corresponding entry of the routing table are updated according to the parameters received with the RREP message as follows: $\text{n}_{\text{hop,D}}$= n_{hop}+1, TimeStamp = Time. Any node that has the $\text{n}_{\text{hop,D}}$field set to a non-infinite value for $\boldsymbol{S}$-$\boldsymbol{D}$ is said to have the *Backward Flag* switched on for that couple. Nodes with both *Forward* and *Backward Flag* on for a given pair are said to be *active* for that couple with sequence number seq given by the corresponding routing table entry. Once the table entry has been written, the node creates a new encrypted RREP message with an updated n_{hop} parameter and broadcasts the message. At the end of the procedure, i.e., when $\boldsymbol{S}$ is reached, the protocol has identified a tunnel connecting $\boldsymbol{S}$ and $\boldsymbol{D}$, composed by the set of active nodes. Such a tunnel is kept active for a fixed time interval after its establishment (tunnel lifetime).

B. Dynamic Data Forwarding

After an active routing tunnel has been established, dynamic transfer of data packets can start. To select the next hop within the tunnel, we propose a CSMA-based distributed contention that exchanges RTS/CTS messages between neighbor nodes. As opposed to other dynamic routing strategies [7], [8], our approach does not rely on any geographic location information. Let $\boldsymbol{X}_n$ be the active node in the tunnel with hop distance n from $\boldsymbol{S}$ that has been selected to forward a DATA packet towards $\boldsymbol{D}$, and let $\mathcal{A}$ be the set of active 1-hop neighbors of $\boldsymbol{X}_n$. To identify the best-suited next-hop $\boldsymbol{X}_{n+1}$, $\boldsymbol{X}_n$broadcasts an encrypted RTS message with parameters S, D, seq, and its hop count distance n_{hop} = $\text{n}_{\text{hop,D}}$. Any node $\boldsymbol{X} \in \mathcal{A}$, which successfully decodes this message and has a hop count $\text{n}_{\text{hop,D}}$less than or equal to the one contained in the RTS, computes a routing metric $m_x(t)$ to assess its adequacy to serve as next-hop, defined as:

$$m_x(t) = (n_{hop,S} + n_{hop,D})^{-1} |h_x(t)|^2 \,. \tag{1}$$

The metric balances two cost criteria: the path length in terms of hop count from $\boldsymbol{S}$ and $\boldsymbol{D}$, and the quality $|h_X(t)|^2$ of the radio link between $\boldsymbol{X}_n$and $\boldsymbol{X}$. $m_x(t)$ increases for nodes with short path length and experiencing good channel conditions to $\boldsymbol{X}_n$. Moreover, the inverse proportionality to $\text{n}_{\text{hop,S}}$+$\text{n}_{\text{hop,D}}$guarantees an advancement towards the destination and avoids routing loops. After computing the metric, the nodes contend with each other to act as data forwarder $\boldsymbol{X}_{n+1}$. For this purpose, each terminal $\boldsymbol{X} \in \mathcal{A}$ sets a backoff timer TIMER_FORWARD, whose duration is inversely proportional to $m_x(t)$. During the backoff period, nodes listen to the medium and compare the received radio signal power to a given threshold value. If the timer expires for a terminal before the sensed power exceeds the threshold, it proposes itself as next-hop by broadcasting an encrypted CTS message with parameters S, D, and seq to its 1-hop neighbors. Otherwise, the node stops the timer because a CTS message has very likely been sent from a neighbor that won the contention. After $\boldsymbol{X}_n$has decoded the CTS message, it unicasts the DATA packet encrypted with its cluster key over the radio channel to the chosen forwarding node $\boldsymbol{X}_{n+1}$. Moreover, a timer TIMER_MONITOR is started that may trigger a repetition of the data forwarding procedure if a successful reception and forwarding of the DATA packet at node $\boldsymbol{X}_{n+1}$ cannot be monitored by node $\boldsymbol{X}_n$within this time interval. If $\boldsymbol{X}_{n+1}$ successfully decodes data, it iterates the dynamic forwarding procedure until the destination is reached. The timer TIMER_MONITOR is reset after $\boldsymbol{X}_n$is able to monitor the transmission of the DATA packet from $\boldsymbol{X}_{n+1}$ to $\boldsymbol{X}_{n+2}$.

C. Reliable and Resilient Data Forwarding

In order to cope with communication errors induced by harsh channel conditions or by misbehaving nodes, we propose a set of additional functionalities for the described protocol. Let $\mathcal{B}$ be the set of active nodes that are common 1-hop neighbors of the two forwarding nodes $\boldsymbol{X}_n$and $\boldsymbol{X}_{n+1}$. Such terminals can overhear packet exchanges between $\boldsymbol{X}_n$and $\boldsymbol{X}_{n+1}$ (RTS/CTS/DATA), as they possess the cluster key for both the nodes. This capability makes it possible for terminals $\boldsymbol{S} \in \mathcal{B}$ to give Xn and Xn+1 support in reliably and resiliently forwarding data packets, hence we refer to them as *support nodes*. In particular, when a support node $\boldsymbol{S}$ overhears a data exchange from $\boldsymbol{X}_n$to $\boldsymbol{X}_{n+1}$, it stores the received packet in a buffer and starts a timer TIMER_MONITOR. As long as the timer is active, $\boldsymbol{S}$ provides on-demand cooperative relaying support to the $\boldsymbol{X}_{n+1}$ and monitors the behavior of this node to detect potential misbehaviors.

Cooperative Relaying: If $\boldsymbol{X}_{n+1}$ fails to decode data from $\boldsymbol{X}_n$, it caches the corrupted version of the packet and sends an encrypted RETransmit REQest (RETREQ) message to its 1-hop neighbors. Support nodes in $\mathcal{B}$ that receive the RETREQ and that successfully decoded the packet sent by $\boldsymbol{X}_n$can give cooperative relaying support by re-encoding and encrypting such data, and forwarding it to $\boldsymbol{X}_{n+1}$. To select the best-suited node, we resort

again to a CSMA-based distributed contention. Each candidate $\boldsymbol{S}$ computes a metric $m_s(t)$ proportional to the channel gain with $\boldsymbol{X}_{n+1}$ estimated from the reception of the RETREQ, and sets a backoff timer TIMER_RELAY inversely proportional to $m_s(t)$. During the backoff phase, carrier sensing is performed. If the sensed power exceeds a given threshold, the node assumes that someone else wins the contention and goes back to its activity. Otherwise, if the time expires, the terminal acts as relay by sending a copy of the cached data to $\boldsymbol{X}_{n+1}$. In this way, two copies of the packet received over two independent channels are available at $\boldsymbol{X}_{n+1}$. Such copies can be jointly decoded using maximum ratio combining, significantly increasing the probability of a successful decoding by taking advantage of spatial diversity [10]. If the reception succeeds, $\boldsymbol{X}_{n+1}$ proceeds with data forwarding to a next-hop node $\boldsymbol{X}_{n+2}$. Otherwise, the terminal sends a retransmit request to $\boldsymbol{X}_n$, that may perform one or more forwarding attempts before triggering a failure.

Local Monitoring: Support nodes can also defend the network against compromised terminals that may entirely or selectively drop packets. As discussed, a node $\boldsymbol{S} \in \mathcal{B}$ that successfully decode data sent by $\boldsymbol{X}_n$starts a TIMER_MONITOR timer. If within this period $\boldsymbol{S}$ decodes a data exchange from $\boldsymbol{X}_{n+1}$ to a next-hop $\boldsymbol{X}_{n+2}$, it verifies in its neighbor list that $\boldsymbol{X}_{n+2}$ is a trusted 2-hop neighbor and compares the incoming data to the original packet received from $\boldsymbol{X}_n$. When the former condition is met and the two packets are identical, $\boldsymbol{S}$ concludes that the communication over the $\boldsymbol{X}_n$- $\boldsymbol{X}_{n+1}$ link succeeded and that $\boldsymbol{X}_{n+1}$ is properly forwarding the payload. If so, the node resets its TIMER_MONITOR timer and goes back to its activity. On the contrary, if the overheard packet is sent to a non-trusted 2-hop neighbor or no data packet transmission is observed before timer expiration, the support node assumes that $\boldsymbol{X}_{n+1}$ maliciously drops packets or redirects them to untrusted nodes. In this case, $\boldsymbol{S}$ resets initiates an alternative path towards the destination through the tunnel by applying the dynamic forwarding algorithm discussed in Section IV-B. Since several support nodes may simultaneously detect the malfunction of $\boldsymbol{X}_{n+1}$, a malicious behavior automatically triggers multi-path forwarding of the data. Duplicate copies of the same packet can easily be suppressed by the destination or by common forwarding nodes in the routing paths. The capability of relying on multi-path data forwarding through the tunnel strongly increases the robustness of the protocol against insider attacks, as will be discussed in Section V. We remark that, as opposed to static multi-path routing protocols [6], our solution introduces multiple flows (thus increasing energy consumption) only if one or several malicious nodes actually hamper the correct operation of the network.

V. Simulation Results

In order to test the effectiveness of the proposed solutions, extensive Omnet++ [12] simulations have been performed. Three routing schemes have been compared: AODV; a basic version of tunnel routing (TR) that does not exploit support nodes (i.e., that implements only the features described in Section IV-B); and the complete Tunnel Routing with Support approach (see Section IV-C), named TRS. All the schemes rely on a carrier-sense based medium access control, modified according to Section IV for the protocols that implement tunnel routing. We have considered topologies composed by 100 nodes spread over a 100m×50m area. Poisson traffic with intensity $\lambda = 0.1$ pk/s is injected in the network by a single source located at (30,25), and addressed to a sink located at (70,25), whereas the other sensor nodes are randomly distributed in the considered region. With this settings, $\boldsymbol{S}$ and $\boldsymbol{D}$ are on average 4 hops away. To analyze the resiliency of the different routing protocols to security attacks, each node within the network can be *malicious* with probability p_{mal}. A malicious terminal participates as usual to route (or tunnel) construction procedures, yet it constantly drops data packets it is supposed to forward. The wireless environment is subject to path loss with exponent $\alpha = 3.5$ and correlated Rayleigh fading. The standard set of parameters used in our simulations is reported in Tab. I. All the results presented in this paper have been obtained by averaging the outcome of 100 independent simulations of duration 10000s, which provided the desired statistical confidence.

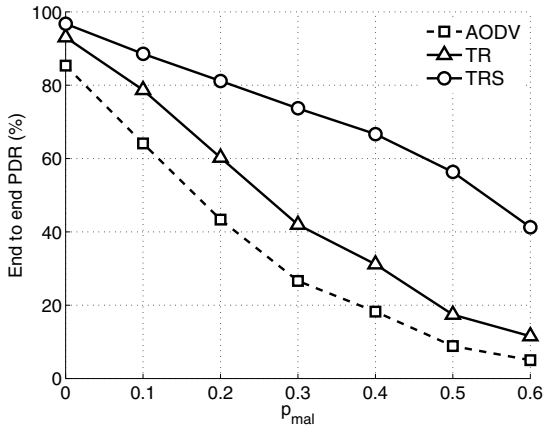

Fig. 1. Dependence of the end-to-end Packet Delivery Ratio on the the fraction of compromised nodes in the network.

The first metric that we consider is the end to end packet delivery ratio (PDR), depicted against p_{mal} in Fig. 1. When nodes in the network properly forward DATA packets (i.e., $p_{mal} = 0$), all the schemes provide high reliability. Moreover, under this hypothesis, protocols that rely on the tunnel routing approach are able to offer an improvement of almost 10% with respect to AODV. This stems from the higher resiliency to channel impairments achieved by the proposed schemes. As discussed, AODV relies on a single path to forward data. Thus, as soon as one of the links along the route incurs harsh channel conditions due to fading or interference, the whole data flow towards the destination collapses until a new path is established. On the contrary, our protocols are able to select a data forwarder among a set of potential candidates, and therefore they manage to avoid weak links in favor of more reliable connections. Fig. 1, also shows that, in the absence of corrupted nodes, TRS slightly improves PDR with respect to basic TR thanks to cooperative relaying, which helps in recovering link failures that may happen despite the dynamic choice of the next hop. Incidentally, we notice that these gains are rather small. This stems from the low number of cooperative retransmissions that are triggered in the topologies under study due to the low level of aggregate interference. The impact of cooperative relaying would raise in networks characterized by multiple simultaneous data flows, whose analysis we regard as part of our future work. Let us now focus on the behavior of the different routing schemes when terminals may maliciously drop data packets. First of all, Fig. 1 shows that the reliability offered by AODV plummets as the number of corrupted nodes in the network grows, with almost no packets delivered for values of p_{mal} higher than 0.5. As malicious terminals actively participate in the route construction procedure, the probability that at least one of such nodes is selected as part of a path identified by AODV increases with p_{mal}. In this condition, static routing is not able to deliver any data to the destination,

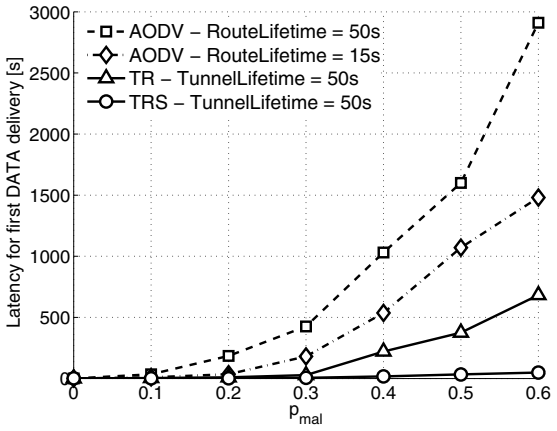

Fig. 2. Time required to deliver the first packet in the network as a function of the fraction of compromised nodes in the network.

since packets constantly get lost as soon as they reach the first compromised terminal along the path. The stall continues until a new route is built, either due to a link failure or to the expiration of the lifetime for the current path, leading to a poor reliability. This effect can be mitigated by using dynamic routing, as shown by Fig. 1, where the TR protocol offers improvements up to 20% over AODV for values of p_{mal} up to 0.3. When TR is implemented, data can still be dropped if a malicious node within the tunnel is selected as a next hop. Nevertheless, since with our approach each frame undergoes a potentially different route to reach the destination, the probability that subsequent packets are dropped is significantly lowered with respect to AODV. Finally, let us consider the performance of TRS. Fig. 1 highlights that our protocol outperforms AODV by as much as 50% in terms of reliability for values of p_{mal} around 0.3. Moreover, the plot shows that even when half of the nodes in the networks are corrupted, TRS is able to successfully deliver 60% of the generated traffic to the sink. This remarkable result stems from the capability of the proposed solution to trigger alternative routes to the destination as soon as support nodes detect a misbehavior by a terminal which is supposed to forward data. In this way, even packets that are dropped along their path have the possibility to be recovered and to eventually reach the destination, thanks to the cooperative behavior of all the nodes within the tunnel.

Another aspect of interest for WSNs subject to insider attacks is the time needed for the first packet to reach the destination, as it describes the capability of a routing scheme to quickly establish a connection and to deliver information to a sink within an hostile environment. The metric is depicted against p_{mal} in Fig. 2. The plot clearly highlights how the latency for the first delivery strongly increases for AODV as the number of compromised nodes in the network grows. This stems once again from the fact that no packet can be delivered to the destination until a route that does not comprise any malicious terminal is identified. On the contrary, the capability offered by tunnel routing schemes to exploit distinct paths within the tunnel when forwarding different packets drastically reduces this latency. Incidentally, we remark that TRS outperforms basic TR thanks to its capability of resorting to multiple routes to recover frames that would otherwise be lost. Starting from this discussion, we observe that the performance in terms of latency for AODV can be improved by forcing the protocol to build routes more frequently (i.e., reducing the Route Lifetime parameter in Tab. I), so that the time required to find a route that does not include any compromised node is shortened. The dash-dotted curve in Fig. 2 shows the

TABLE I
PARAMETERS USED IN OUR SIMULATIONS

Tx power / Noise Floor / CS threshold	-90 dBm, 92 dBm
Data Rate	250 kbit/s
Short Retry Limit at MAC layer	3
Payload length / Control pkts	1024 bit, 112 bit
Route lifetime	50 s

results obtained following this approach. Two remarks can be made. First of all, although reduced, the latency undergone by AODV to deliver the first packet is still considerably higher that the ones obtained when using tunnel routing. Second, an increase in the frequency of route constructions induces a significant raise in the terms of control overhead, and thus of energy consumption, which is a metric of capital importance in battery constrained WSNs, since a lower energy consumption turns into a longer network lifetime. Such a trend was confirmed by our simulations, whose results are not reported here due to space constraints.

In conclusion, we can infer that not only does TRS guarantee high end to end delivery reliability even when a large fraction of the network is compromised, but also it is able to connect source and destination with low latency and at low costs in terms of energy consumption with respect to existing and widely implemented static routing solutions.

VI. Conclusions

In this paper, we have introduced dynamic tunnel routing, that enables efficient and resilient data forwarding in WSNs. Tunnel routing exploits a route construction procedure to identify a family of trusted sensor nodes that form a tunnel between a source and a destination and that later contend among each other to act as data forwarders. Moreover, terminals within the tunnel may act as support nodes by providing on-demand re-transmissions of data packets to combat channel impairments, or by initiating multi-path data forwarding towards the destination in case they detect a misbehavior caused by a selective forwarding attack. Simulation results have shown the advantages of the proposed solution with respect to existing static routing protocols.

References

[1] C. Perkins and E. Royer, "Ad-hoc On Demand Distance Vector Routing," in *IEEE WMCSA*, New Orleans, LA, US, Feb. 1999.
[2] D. Johnson, D. Maltz, and J. Broch, *DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks*, ser. Ad Hoc Networking. Addison-Wesley, 2001, ch. 5, pp. 139–172.
[3] S. Mueller, R. P. Tsang, and D. Ghosal, *Multipath Routing in Mobile Ad Hoc Networks: Issues and Challenges*, ser. Lecture Notes in Computer Science. Springer Berlin, 2004, pp. 209–234.
[4] S. Zhu, S. Setia, and S. Jajodia, "LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks," in *ACM CCS*, 2003.
[5] C. Karlof and D. Wagner, "Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures," in *IEEE GlobeCom*, 2003.
[6] B. Deb, S. Bhatnagar, and B. Nath, "ReInForM: Reliable Information Forwarding using Multiple Paths in Sensor Networks," in *IEEE LCN*, 2003.
[7] M. Zorzi and R. R. Rao, "Geographic Random Forwarding (GeRaF) for Ad Hoc and Sensor Networks: Multihop Performance," *IEEE Transactions on Mobile Computing*, vol. 2, no. 4, pp. 337–348, 2003.
[8] P. Coronel, R. Doss, and W. Schott, "Geographic Routing with Cooperative Relaying and Leapfrogging in Wireless Sensor Networks," in *IEEE GlobeCom*, Washington DC, US, Nov. 26-30 2007.
[9] A. Munari and W. Schott, "Performance Assessment of a Class of Cross-Layer Optimized Protocols for Geographic Routing," in *IEEE PIMRC*, Cannes, FR, Sep. 15–18 2008.
[10] J. Laneman, D. Tse, and G. Wornell, "Cooperative Diversity in Wireless Networks: Efficient Protocol Design and Outage Behaviour," *IEEE Transactions on Information Theory*, vol. 50, no. 12, pp. 3062–3080, 2004.
[11] S. Lee and Y. Choi, "A Resilient Packet Forwarding Scheme Against Malicious Packet-Dropping nodes in Sensor Networks," in *SANS*, 2006.
[12] A. Varga, "Omnet++," http://www.omnetpp.org, 2001.