International Journal of Computer Science
and Engineering (IJCSE)
ISSN(P): 2278-9960; ISSN(E): 2278-9979
Vol. 3, Issue 2, Mar 2014, 111-126
© IASET

International Academy of Science,
Engineering and Technology
IASET Connecting Researchers; Nurturing Innovations

MITIGATING A TRUST-AWARE ROUTING FRAMEWORK FOR CATCHING PACKET MODIFIERS IN WSNs

MADHIRALA CHANDRAKALA[1] & D. KOTESWARA RAO[2]

[1]PG Student, Department of Computer Science and Engineering, Chadalawada Ramanamma Engineering College,
Tirupathi, Andhra Pradesh, India

[2]Associate Professor, Department of Computer Science and Engineering, Chadalawada Ramanamma Engineering College,
Tirupathi, Andhra Pradesh, India

ABSTRACT

Packet dropping and modification are common attacks that can be launched by an adversary to disrupt communication in wireless multihop sensor networks. Many schemes have been proposed to mitigate or tolerate such attacks, but very few can effectively and efficiently identify the intruders. To address this problem, to secure the WSNs against adversaries misdirecting the multihop routing, we have designed and implemented TARF, a robust trust-aware routing framework for dynamic WSNs. Without tight time synchronization or known geographic information, TARF provides trustworthy and energy-efficient route. Most importantly, TARF proves effective against those harmful attacks developed out of identity deception; the resilience of TARF is verified through extensive evaluation with both simulation and empirical experiments on large-scale WSNs under various scenarios including mobile and RF-shielding network conditions. Further, we have implemented a low-overhead TARF module in Tiny OS; as demonstrated, this implementation can be incorporated into existing routing protocols with the least effort. Based on TARF, we also demonstrated a proof-of-concept mobile target detection application that functions well against an anti detection mechanism.

KEYWORDS: Wireless Sensor Networks, Routing Protocols, Security

1. INTRODUCTION

In a wireless sensor network, sensor nodes monitor the environment, detect events of interest, produce data, and collaborate in forwarding the data toward a sink, which could be a gateway, base station, storage node, or querying user. Because of the ease of deployment, the low cost of sensor nodes and the capability of self-organization, a sensor network is often deployed in an unattended and hostile environment to perform the monitoring and data collection tasks. When it is deployed in such an environment, it lacks physical protection and is subject to node compromise. After compromising one or multiple sensor nodes, an adversary may launch various attacks to disrupt the in-network communication. Among these attacks, two common ones are dropping packets and modifying packets, i.e., compromised nodes drop or modify the packets that they are supposed to forward.

To deal with packet droppers, a widely adopted countermeasure is multipath forwarding in which each packet is forwarded along multiple redundant paths and hence packet dropping in some but not all of these paths can be tolerated. To deal with packet modifiers, most of existing countermeasures aim to filter modified messages en-route within a certain number of hops. These countermeasures can tolerate or mitigate the packet dropping and modification attacks, but the intruders are still there and can continue attacking the network without being caught. To locate and identify packet droppers and modifiers, it has been introduced that nodes continuously monitor the forwarding behaviors of their neighbors to determine if their neighbors are misbehaving, and the approach can be extended by using the reputation based mechanisms to allow nodes to infer whether a non neighbor node is trustable. This methodology may be subject to high-energy cost incurred by the promiscuous operating mode of wireless interface; moreover, the reputation mechanisms have to be exercised with cautions to avoid or mitigate bad mouth attacks and others.

Recently, Ye et al. introduced a probabilistic nested marking (PNM) scheme. But with the PNM scheme, modified packets should not be filtered out en route because they should be used as evidence to infer packet modifiers; hence, it cannot be used together with existing packet filtering schemes.

As a harmful and easy-to-implement type of attack, a malicious node simply replays all the outgoing routing packets from a valid node to forge the latter node's identity; the malicious node then uses this forged identity to participate in the network routing, thus disrupting. The network traffic. Those routing packets, including their original headers, are replayed without any modification. Even if this malicious node cannot directly overhear the valid node's wireless transmission, it can collude with other malicious nodes to receive those routing packets and replay them somewhere far away from the original valid node, which is known as a wormhole attack. Since a node in a WSN usually relies solely on the packets received to know about the sender's identity, replaying routing packets allows the malicious node to forge the identity of this valid node.

After "stealing" that valid identity, this malicious node is able to misdirect the network traffic. For instance, it may drop packets received, forward packets to another node not supposed to be in the routing path, or even form a transmission loop through which packets are passed among a few malicious nodes infinitely.

It is often difficult to know whether a node forwards received packets correctly even with overhearing techniques. Sinkhole attacks are another kind of attacks that can be launched after stealing a valid identity. In a sinkhole attack, a malicious node may claim.

Itself to be a base station through replaying all the packets from a real base station.

Honest node with transient failure. Without proper protection, WSNs with existing routing protocols can be completely devastated under certain circumstances. In an emergent sensing application through WSNs, saving the network from being devastated becomes crucial to the success of the application.

Unfortunately, most existing routing protocols for WSNs either assume the honesty of nodes and focus on energy efficiency, or attempt to exclude unauthorized participation by encrypting data and authenticating packets. Examples of these encryption and authentication schemes For WSNs include TinySec, Spins, TinyPK, and TinyECC. Admittedly, it is important to consider efficient energy use for battery-powered Sensor nodes and the robustness of routing under topological changes as well as common faults in a wild environment. However, it is also critical to incorporate security as one of the most important goals; meanwhile, even with perfect encryption and authentication, by replaying routing information, a malicious node can still participate in the network using another valid node's identity. The gossiping-based routing protocols offer certain protection against attackers by selecting random neighbors to forward packets, but at a price of considerable overhead in propagation time and energy use. In addition to the cryptographic methods, trust and reputation management has been employed in generic adhoc networks and WSNs to secure routing protocols.

Basically, a system of trust and reputation management assigns each node a trust value according to its past performance in routing. Then, such trust values are used to help decide a secure and Efficient route.

However, the proposed trust and reputation management systems for generic adhoc networks target only relatively powerful hardware platforms such as laptops and smart phones. Those systems cannot be applied to WSNs due to the excessive overhead for resource-constrained sensor nodes powered by batteries. As far as WSNs are concerned, secure routing solutions based on trust and reputation management rarely address the identity deception through replaying routing information.

The countermeasures proposed so far strongly depends on either tight time synchronization or known geographic information while

Their effectiveness against attacks exploiting the replay of routing Information has not been examined yet. At this point, to protect WSNs from the harmful attacks exploiting the replay of routing information, we have designed and implemented a robust trust-aware routing framework, TARF, to secure routing solutions in wireless sensor networks. Based on the unique characteristics of resource-constrained WSNs, the design of TARF centers on trustworthiness and energy efficiency. Though TARF can be developed into a complete and independent routing protocol, the purpose is to allow existing routing protocols to incorporate our implementation of TARF with the least effort and thus producing a secure and efficient fully functional protocol.

Unlike other security measures, TARF requires neither tight time synchronization nor known geographic information. Most importantly, TARF proves resilient under various attacks exploiting the replay of routing information, which is not achieved by previous security protocols. Even under strong attacks such as sinkhole attacks, wormhole attacks as well as Sybil attacks, and hostile mobile network condition, TARF demonstrates steady improvement in network performance. The effectiveness of TARF is verified through extensive evaluation with simulation and empirical experiments on large-scale WSNs. Finally, we have implemented a ready-to-use TARF module with low overhead, which as demonstrated can be integrated into existing routing protocols with ease; the demonstration of a proof- considerations of TARF in Section 2. Then, we elaborate the design of TARF in Section 3, including the routing procedure as well as the Energy- Watcher and Trust Manager components. In Section 4, we present the simulation results of TARF against various attacks through replaying routing information in static, mobile and RF-shielding conditions. Section 5 further presents the implementation of TARF, empirical evaluation at a large sensor network and a resilient proof-of-concept mobile target detection application based on TARF.

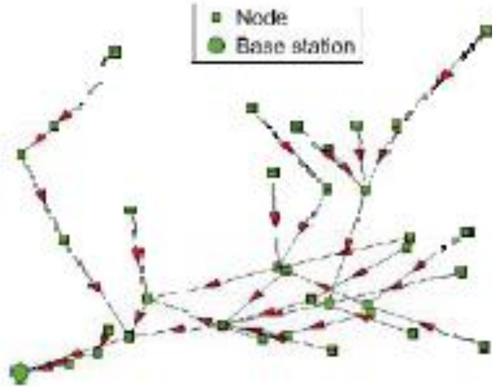


Figure 1: Multihop Routing for Data Collection of a WSN

2. SYSTEM MODEL

2.1 Network Assumptions

We consider a typical deployment of sensor networks, where a large number of sensor nodes are randomly deployed in a two dimensional area. Each sensor node generates sensory data periodically and all these nodes collaborate to forward packets containing the data toward a sink. The sink is located within the network. We assume all sensor nodes and the sink are loosely time synchronized, which is required by many applications. Attack resilient time synchronization schemes, which have been widely investigated in wireless sensor networks, can be employed. The sink is aware of the network topology, which can be achieved by requiring nodes to report their neighboring nodes right after deployment.

2.2 Security Assumptions and Attack Model

We assume the network sink is trustworthy and free of compromise, and the adversary cannot successfully compromise regular sensor nodes during the short topology establishment phase after the network is deployed. This assumption has been widely made in existing work. After then, the regular sensor nodes can be compromised. Compromised nodes mayor may not collude with each other.

A compromised node can launch the following two attacks:

Packet Dropping: A compromised node drops all or some of the packets that is supposed to forward. It may also drop the data generated by itself for some malicious purpose such as framing innocent nodes.

Packet Modification: A compromised node modifies all or some of the packets that is supposed to forward. It may also modify the data it generates to protect itself from being identified or to accuse other nodes.

3. THE EXISTING SCHEME

Our scheme consists of a system initialization phase and several equal-duration rounds of intruder identification phases.

In the Initialization Phase: Sensor nodes form a topology which is a directed acyclic graph (DAG). A routing tree is extracted from the DAG. Data reports follow the routing tree structure.

In Each Round: Data are transferred through the routing tree to the sink. Each packet sender/ forwarder adds a small number of extra bits to the packet and also encrypts the packet. When one round finishes, based on the extra bits carried in the received packets, the sink runs a node categorization algorithm to identify nodes that must be bad (i.e., packet droppers or modifiers) and nodes that are suspiciously bad (i.e., suspected to be packet droppers and modifiers).

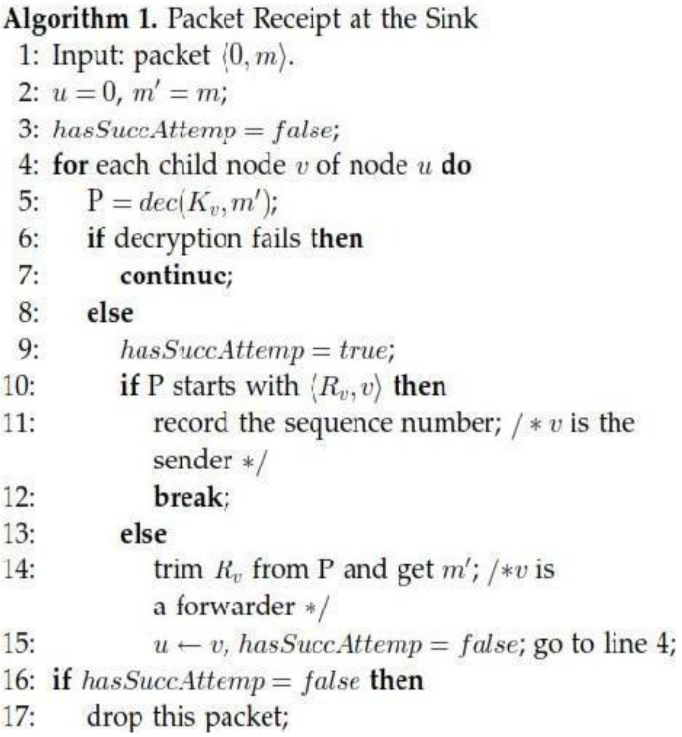
The Routing Tree is Reshaped Every Round: As a certain number of rounds have passed, the sink will have collected information about node behaviors in different routing topologies. The information includes which nodes are bad for sure, which nodes are suspiciously bad, and the nodes' topological relationship. To further identify bad nodes from the potentially large number of suspiciously bad nodes, the sink runs heuristic ranking algorithms.

In the following sections, we first present the algorithm for DAG establishment and packet transmission, which is followed by our proposed categorization algorithm, tree structure reshaping algorithm, and heuristic ranking algorithms. To ease the presentation, we first concentrate on packet droppers and assume no node collusion. After that, we present how to extend the presented scheme to handle node collusion and detect packet modifiers, respectively.

3.1 DAG Establishment and Packet Transmission

All sensor nodes form a DAG and extract a routing tree from the DAG. The sink knows the DAG and the routing tree, and shares a unique key with each node. When a node wants to send out a packet, it attaches to the packet a sequence number, encrypts the packet only with the key shared with the sink, and then forwards the packet to its parent on the routing tree. When an innocent intermediate node receives a packet, it attaches a few bits to the packet to mark the forwarding path of the packet, encrypts the packet, and then forwards the packet to its parent. On the contrary, a misbehaving intermediate node may drop a packet it receives. On receiving a packet, the sink decrypts it, and thus finds out the original sender and the packet sequence number. The sink tracks the sequence numbers of received packets for every node, and for every certain time interval, which we call a round, it calculates the packet dropping ratio for every node. Based on the dropping ratio and the knowledge of the topology, the sink identifies packet droppers based on rules we derive.

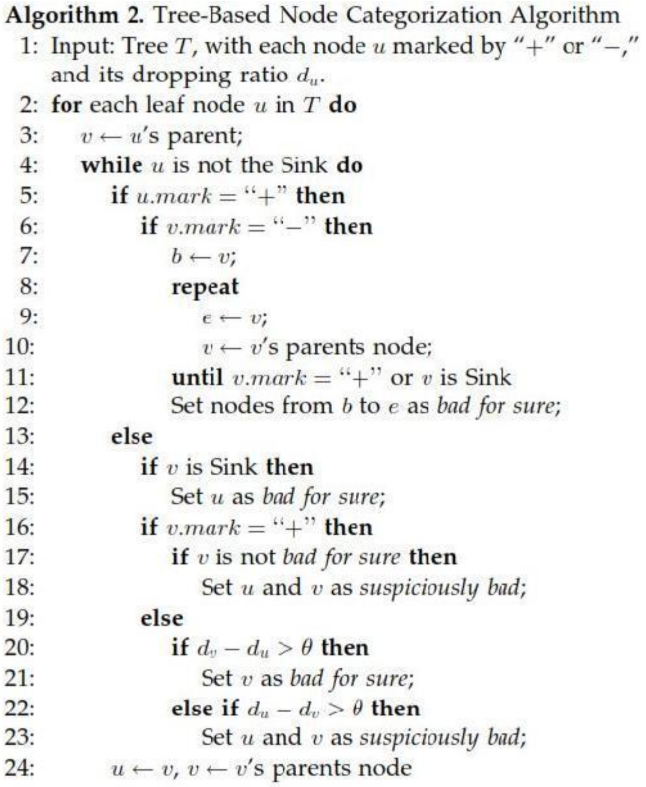
The process of packet receipt at the sink can be formalized as Algorithm 1. An example is provided in the supplementary file, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2011.117. In this example, we show how the packet is sent and forwarded as well as how the packet is processed by the sink.



Algorithm 1. Packet Receipt at the Sink

```
 1: Input: packet $\langle 0, m \rangle$.
 2: $u = 0$, $m' = m$;
 3: $hasSuccAttemp = false$;
 4: for each child node $v$ of node $u$ do
 5:     P $= dec(K_v, m')$;
 6:     if decryption fails then
 7:         continuc;
 8:     else
 9:         $hasSuccAttemp = true$;
10:         if P starts with $\langle R_v, v \rangle$ then
11:             record the sequence number; $/ * v$ is the
                sender $* /$
12:             break;
13:         else
14:             trim $R_v$ from P and get $m'$; $/ * v$ is
                a forwarder $* /$
15:             $u \leftarrow v$, $hasSuccAttemp = false$; go to line 4;
16: if $hasSuccAttemp = false$ then
17:     drop this packet;
```

Based on the rules, we develop a node categorization algorithm to find nodes that are bad for sure or suspiciously bad. The formal algorithm is presented in Algorithm 2.



Algorithm 2. Tree-Based Node Categorization Algorithm

```
Input: Tree $T$, with each node $u$ marked by "+" or "−," and its dropping ratio $d_u$.
for each leaf node $u$ in $T$ do
    $v \leftarrow u$'s parent;
    while $u$ is not the Sink do
        if $u.mark =$ "+" then
            if $v.mark =$ "−" then
                $b \leftarrow v$;
                repeat
                    $\epsilon \leftarrow v$;
                    $v \leftarrow v$'s parents node;
                until $v.mark =$ "+" or $v$ is Sink
                Set nodes from $b$ to $e$ as bad for sure;
        else
            if $v$ is Sink then
                Set $u$ as bad for sure;
            if $v.mark =$ "+" then
                if $v$ is not bad for sure then
                    Set $u$ and $v$ as suspiciously bad;
            else
                if $d_v - d_u > \theta$ then
                    Set $v$ as bad for sure;
                else if $d_u - d_v > \theta$ then
                    Set $u$ and $v$ as suspiciously bad;
        $u \leftarrow v$, $v \leftarrow v$'s parents node
```

4. DESIGN CONSIDERATIONS

Before elaborating the detailed design of TARF, we would like to clarify a few design considerations first, including certain assumptions and the goals.

4.1 Assumptions

We target secure routing for data collection tasks, which are one of the most fundamental functions of WSNs. In a data collection task, a sensor node sends its sampled data to a remote base station with the aid of other intermediate nodes, as shown in Figure 1. Though there could be more than one base station, our routing approach is not affected by the number of base stations; to simplify our discussion, we assume that there is only one base station. An adversary may forge the identity of any legal node through replaying that node's outgoing routing packets and spoofing the acknowledgment packets, even remotely through a wormhole.

Additionally, to merely simplify the introduction of TARF, we assume no data aggregation is involved. None-the less, our approach can still be applied to cluster-based WSNs with static clusters, where data are aggregated by clusters before being relayed . Cluster-based WSNs allows for the great savings of energy and bandwidth through aggregating data from children nodes and per forming routing and transmission for children nodes. In a cluster-based WSN, the cluster headers themselves form a sub network; after certain data reach a cluster header, the aggregated data will be routed to a base station only through such a sub network consisting of the cluster headers. Our framework can then be applied to this sub network to achieve secure routing for cluster-based WSNs. TARF may run on cluster headers only and the cluster headers communicate with their children nodes directly since a static cluster has known relationship between a cluster header and its children nodes, though any link-level security features may be further employed.

Finally, we assume a data packet has at least the following fields: the sender id, the sender sequence number, the next-hop node id (the receiver in this one-hop transmis sion), the source id (the node that initiates the data), and the source's sequence number. We insist that the source node's information should be included for the following reasons because that allows the base station to track whether a data packet is delivered. It would cause too much overhead to transmit all the one-hop information to the base station. Also, we assume the routing packet is sequenced.

4.2 Authentication Requirements

Though a specific application may determine whether data encryption is needed, TARF requires that the packets are properly authenticated, especially the broadcast packets from the base station. The broadcast from the base station is asymmetrically authenticated so as to guarantee that an adversary is not able to manipulate or forge a broadcast message from the base station at will. Importantly, with authenticated broadcast, even with the existence of attack ers, TARF may use Trust Manager and the received broadcast packets about delivery information to choose trustworthy path by circumventing compromised nodes. Without being able to physically capturing the base station, it is generally very difficult for the adversary to manipulate the base station broadcast packets which are asymmetrically authenticated. The asymmetric authentication of those broadcast packets from the base station is crucial to any successful secure routing protocol. It can be achieved through existing asymmetrically authenticated broadcast schemes that may require loose time synchronization. As an example, TESLA achieves asymmetric authenticated broadcast through a symmetric cryptographic algorithm and a loose delay schedule to disclose the keys from a key chain. Other examples of asymmetric authenticated broadcast schemes requiring either loose or no time synchronization.

Considering the great computation cost incurred by *a* strong asymmetric authentication scheme and the difficulty in key management, a regular packet other than a base station broadcast packet may only be moderately authenticated through existing symmetric schemes with a limited set of keys, such as the message authentication code provided by Tiny Sec. It is possible that an adversary physically captures a nonbase legal node and reveals its key for the symmetric authentication. With that key, the adversary can forge the identity of that non base legal node and joins the network "legally." However, when the adversary uses its fake identity to falsely attract a great amount of traffic, after receiving broadcast packets about delivery information, other legal nodes that directly or indirectly forwards packets through it will start to select a more trustworthy path through Trust Manager.

4.3 Goals

TARF mainly guards a WSN against the attacks misdirecting the multihop routing, especially those based on identity theft through replaying the routing information. This paper does not address the denial-of-service (DoS) [3] attacks, where an attacker intends to damage the network by exhausting its resource. For instance, we do not address the DoS attack of congesting the network by replaying numerous packets or physically jamming the network. TARF aims to achieve the following desirable properties:

High throughput. Throughput is defined as the ratio of the number of all data packets delivered to the base station to the number of all sampled data packets. In our evaluation, throughput at a moment is computed over the period from the beginning time (0) until that particular moment. Note that single-hop retransmission may happen, and that duplicate packets are considered as one packet as far as throughput is concerned. Throughput reflects how efficiently the network is collecting and delivering data. Here, we regard high throughput as one of our most important goals.

Energy efficiency. Data transmission accounts for a major portion of the energy consumption. We evaluate energy efficiency by the average energy cost to successfully deliver a unit-sized data packet from a source node to the base station. Note that link-level retransmission should be given enough attention when considering energy cost since each retransmission causes a noticeable increase in energy consumption. If every node in a WSN consumes approximately the same energy to transmit a unit-sized data packet, we can use another metric hop-per-delivery to evaluate energy efficiency. Under that assumption, the energy consumption depends on the number of hops, i.e., the number of one-hop transmissions occurring. To evaluate how efficiently energy is used, we can measure the average hops that each delivery of a data packet takes, abbreviated as hop-per-delivery.

Scalability and adaptability. TARF should work well with WSNs of large magnitude under highly dynamic contexts. We will evaluate the scalability and adaptability of TARF through experiments with large-scale WSNs and under mobile and hash network conditions.

Here, we do not include other aspects such as latency, load balance, or fairness. Low latency, balanced network load, and good fairness requirements can be enforced in specific routing protocols incorporating TARF.

5. DESIGN OF TARF

TARF secures the multihop routing in WSNs against intruders misdirecting the multihop routing by evaluating the trustworthiness of neighboring nodes. It identifies such intruders by their low trustworthiness and routes data through paths circumventing those intruders to achieve satisfactory throughput. TARF is also energy efficient, highly scalable, and well adaptable. Before introducing the detailed design, we first introduce several necessary notions here. Neighbor. For a node N, a neighbor (neighboring node) of N is a node that is reachable from N with one-hop wireless transmission.

Trust level. For a node N, the trust level of a neighbor is a decimal number in [0, 1], representing N's opinion of that neighbor's level of trustworthiness. Specifically, the trust level of the neighbor is N's estimation of the probability that this neighbor correctly delivers data received to the base station. That trust level is denoted as T in this paper.

Energy cost. For a node N, the energy cost of a neighbor is the average energy cost to successfully deliver a unit-sized data packet with this neighbor as its next-hop node, from N to the base station. That energy cost is denoted as E in this paper.

5.1 Overview

For a TARF-enabled node N to route a data packet to the base station, N only needs to decide to which neighboring node it should forward the data packet considering both the trustworthiness and the energy efficiency. Once the data packet is forwarded to that next-hop node, the remaining task to deliver the data to the base station is fully delegated to it, and N is totally unaware of what routing decision its next-hop node makes. N maintains a neighborhood table with trust level values and energy cost values for certain known neighbors. It is sometimes necessary to delete some neighbors' entries to keep the table size acceptable. The technique of maintaining a neighborhood table of a moderate size is demonstrated by Woo et al.; TARF may employ the same technique.

In TARF, in addition to data packet transmission, there are two types of routing information that need to be exchanged: broadcast messages from the base station about data delivery and energy cost report messages from each node. Neither message needs acknowledgment. A broadcast message from the base station is flooded to the whole network.

The freshness of a broadcast message is checked through its field of source sequence number. The other type of exchanged routing information is the energy cost report message from each node, which is broadcast to only its neighbors once. Any node receiving such an energy cost report message will not forward it.

For each node N in a WSN, to maintain such a neighborhood table with trust level values and energy cost values for certain known neighbors, two components, Energy Watcher and Trust Manager, run on the node (Figure 2). Energy Watcher is responsible for recording the energy cost for each known neighbor, based on N's observation of one-hop transmission to reach its neighbors and the energy cost report from those neighbors. A compromised node may falsely report an extremely low energy cost to lure its neighbors into selecting this compromised node as their next-hop node; however, these TARF-enabled neighbors eventually abandon that compromised next-hop node based on its low trustworthiness as tracked by Trust Manager. Trust Manager is responsible for tracking trust level values of neighbors based on network loop discovery and broadcast messages from the base station about data delivery. Once N

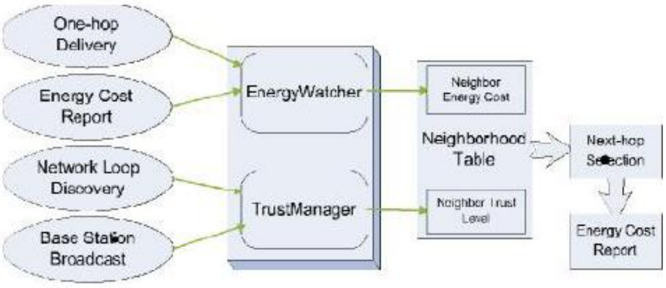


Figure 2

Figure 2, Each node selects a next-hop node based on its neighborhood table, and broadcast its energy cost within its neighborhood. To maintain this neighborhood table, Energy Watcher and Trust Manager on the node keep track of related events (on the left) to record the energy cost and the trust level values of its neighbors. is able to decide its next-hop neighbor according to its neighborhood table, it sends out its energy report message: it broadcasts to all its neighbors its energy cost to deliver a packet from the node to the base station. The energy cost is computed as in Section 3.3 by Energy Watcher. Such an energy cost report also serves as the input of its receivers' Energy Watcher.

5.2 Routing Procedure

TARF, as with many other routing protocols, runs as a periodic service. The length of that period determines how frequently routing information is exchanged and updated. At the beginning of each period, the base station broadcasts a message about data delivery during last period to the whole network consisting of a few contiguous packets (one packet may not hold all the information). Each such packet has a field to indicate how many packets are remaining to complete the broadcast of the current message. The completion of the base station broadcast triggers the exchange of energy report in this new period. Whenever a node receives such a broadcast message from the base station, it knows that the most recent period has ended and a new period has just started.

No tight time synchronization is required for a node to keep track of the beginning or ending of a period. During each period, the Energy Watcher on a node monitors energy consumption of one-hop transmission to its neighbors and processes energy cost reports from those neighbors to maintain energy cost entries in its neighborhood table; its Trust Manager also keeps track of network loops and processes broadcast messages from the base station about data delivery to maintain trust level entries in its neighborhood table.

To maintain the stability of its routing path, a node may retain the same next-hop node until the next fresh broadcast message from the base station occurs. Meanwhile, to reduce traffic, its energy cost report could be configured to not occur again until the next fresh broadcast message from the base station. If a node does not change its next-hop node selection until the next broadcast message from the base station, that guarantees all paths to be loop-free, as can be deducted from the procedure of next-hop node selection. However, as noted in our experiments, that would lead to slow improvement in routing paths. Therefore, we allow a node to change its next-hop selection in a period when its current next-hop node performs the task of receiving and delivering data poorly.

Next, we introduce the structure and exchange of routing information as well as how nodes make routing decisions in TARF.

5.2.1 Structure and Exchange of Routing Information

A broadcast message from the base station fits into at most a fixed small number of packets. Such a message consists of some pairs of <node id of a source node, an undelivered sequence interval [a, b] with a significant length>, <node id of a source node, minimal sequence number received in last period, maximum sequence number received in last period>, as well as several node id intervals of those without any delivery record in last period. To reduce overhead to an acceptable amount, our implementation selects only a limited number of such pairs to broadcast and proved effective. Roughly, the effectiveness can be explained as follows: the fact that an attacker attracts a great deal of traffic from many nodes often gets revealed by at least several of those nodes being deceived with a high likelihood. The undelivered sequence interval [a, b] is explained as follows: the base station searches the source sequence numbers received in last period, identifies which source sequence numbers for the source node with this id are missing, and chooses certain significant interval [a, b] of missing source sequence numbers as an undelivered sequence interval. For example, the base station may have all the source sequence numbers for the source node 2 as in last period. Then, is an undelivered sequence interval; is also recorded as the sequence boundary of delivered packets. Since the base station is usually connected to a powerful platform such as a desktop, a program can be developed on that powerful platform to assist in recording all the source sequence numbers and finding undelivered sequence intervals.

Accordingly, each node in the network stores a table of <node id of a source node, a forwarded sequence interval [a, b] with a significant length> about last period. The data packets with the source node and the sequence numbers falling in this forwarded sequence interval [a, b] have already been forwarded by this node. When the node receives a broadcast message about data delivery, its Trust Manager will be able to identify which data packets forwarded by this node are not delivered to the base station. Considering the overhead to store such a table, old entries will be deleted once the table is full. Once a fresh broadcast message from the base station is received, a node immediately invalidates all the existing energy cost entries: it is ready to receive a new energy report from its neighbors and choose its new next-hop node afterward. Also, it is going to select a node either after a timeout is reached or after it has received an energy cost report from some highly trusted candidates with acceptable energy cost. A node immediately broadcasts its energy cost to its neighbors only after it has selected a new next-hop node. That energy cost is computed by its Energy Watcher. A natural question is which node starts reporting its energy cost first. For that, note that when the base station is sending a broadcast message, a side effect is that its neighbors receiving that message will also regard this as an energy report: the base station needs 0 amount of energy to reach itself. As long as the original base station is faithful, it will be viewed as a trustworthy candidate by Trust Manager on the neighbors of the base station. There- fore, those neighbors will be the first nodes to decide their next-hop node, which is the base station; they will start reporting their energy cost once that decision is made.

5.2.2 Route Selection

Now, we introduce how TARF decides routes in a WSN. Each node N relies on its neighborhood table to select an optimal route, considering both energy consumption and reliability. TARF makes good efforts in excluding those nodes that misdirect traffic by exploiting the replay of routing information. For a node N to select a route for delivering data to the base station, N will select an optimal next-hop node from its neighbors based on trust level and energy cost and forward the data to the chosen next-hop node immediately. The neighbors with trust levels below a certain threshold will be excluded from being considered as candidates. Among the remaining known neighbors, N will select its next-hop node through evaluating each neighbor b based on a tradeoff between T_{Nb} and^ , with E_{Nb} and T_{Nb} being b's energy cost and trust level value in the neighborhood table, respectively, (see Sections 3.3, 3.4). Basically, ENI, reflects the energy cost of delivering a packet to the base station from N assuming that all the nodes in the route are honest;

TST approximately reflects the number of the needed value of ^ even with a low T_{Nb}. Therefore, TARF prefers nodes with significantly higher trust values; this preference of trustworthiness effectively protects the network from an adversary who forges the identity of an attractive node such as a base station. For deciding the next-hop node, a specific tradeoff between TNI, and ^ is demonstrated.

Observe that in an ideal misbehavior-free environment, all nodes are absolutely faithful, and each node will choose a neighbor through which the routing path is optimized in terms of energy; thus, an energy-driven route is achieved.

5.3 Energy Watcher

Here, we describe how a node N's Energy Watcher computes the energy cost E^ for its neighbor b in N's neighborhood table and how N decides its own energy cost EN . Before going further, we will clarify some notations. E_{Nb} mentioned is the average energy cost of successfully delivering a unit-sized data packet from N to the base station, with b as N's next-hop node being responsible for the remaining route. Here, one-hop retransmission may occur until the acknowledgment is received or the number of retransmissions reaches a certain threshold. The cost caused by one hop retransmissions should be included when computing Eu_b . Suppose N decides that A should be its next-hop node after comparing energy cost and trust level. Then, N's energy cost is EN !4 EN A. Denote Ew-b as the average energy cost of successfully delivering a data packet from N to its neighbor b with one hop. Note that the retransmission cost needs to be considered. With the above notations, it is straightforward to establish the following relation:

$$E_{Nb} \ ¼ \ E_{N-b} + E_{b} .$$

Since each known neighbor b of N is supposed to broadcast its own energy cost E_b to N , to compute ENI> , N still needs to know the value $E_{N\cdot b}$, i.e., the average energy cost of successfully delivering a data packet from N to its neighbor b with one hop. For that, assuming that the endings (being acknowledged or not) of one-hop transmissions from N to b are independent with the same probability p_{succ} of being acknowledged, we first compute the average number of one-hop sendings needed before the acknowledgment is received as follows:

$$\sum_{i=1}^{\infty} i \cdot p_{succ} \cdot (1 - p_{succ})^{i-1} - \frac{1}{p_{succ}} .$$

The remaining job for computing ENb is to get the probability p_{succ} that a one-hop transmission is acknowledged. Considering the variable wireless connection among wireless sensor nodes, we do not use the simplistic averaging method to compute p_{succ} . Instead, after each transmission from N to b, N 's Energy Watcher will update p_{succ} based on whether that transmission is acknowledged or not with a weighted averaging technique. We use a binary variable Ack to record the result of current transmission: 1 if an acknowledgment is received; otherwise, 0. Given Ack and the last probability value of an acknowledged transmission $p_{oW\ succ}$, an intuitive way is to use a simply weighted average of Ack and $p_{old\ succ}$ as the value of $p_{new\ succ}$. That is what is essentially adopted in the aging mechanism. However, that method used against sleeper attacks still suffers periodic attacks [30]. To solve this problem, we update the p_{succ} value using two different weights as in our previous work, a relatively big Wdegrade €(0,1) and a relatively small $w_{upgrade}$ €(0,1) as follows:

$$p_{new\_succ} = \begin{cases} (1 - w_{degrade}) \times p_{old\_succ} + w_{degrade} \times Ack, \\ \text{if } Ack = 0. \\ (1 - w_{upgrade}) \times p_{old\_succ} + w_{upgrade} \times Ack, \\ \text{if } Ack = .1. \end{cases}$$

The two parameters Wdegrade ^{a}"d w_{upgra}de allow flexible application requirements. $w_{degradl}$, and w_{upglad}e represent the extent to which upgraded and degraded performance are rewarded and penalized, respectively. If any fault and compromise is very likely to be associated with a high risk, Wdegrade should be assigned a relatively high value to penalize fault and compromise relatively heavily; if a few positive transactions can't constitute evidence of good connectivity which requires many more positive transactions, then Wupgrade should be assigned a relatively low value.

5.4 Trust Manager

A node N's Trust Manager decides the trust level of each neighbor based on the following events: discovery of network loops, and broadcast from the base station about data delivery. For each neighbor b of N, T [vj_b denotes the trust level of b in N's neighborhood table. At the beginning, each neighbor is given a neutral trust level 0.5. After any of those events occurs, the relevant neighbors' trust levels are updated.

Note that many existing routing protocols have their own mechanisms to detect routing loops and to react accordingly [31], [32], [28]. In that case, when integrating TARF into those protocols with anti loop mechanisms, Trust Manager may solely depend on the broadcast from the base station to decide the trust level; we adopted such a policy when implementing TARF. If anti loop mechanisms are both enforced in the TARF. then the resulting hybrid protocol may overly react toward the discovery of loops. Though sophisticated loop-discovery methods exist in the currently developed protocols, they often rely on the comparison of specific routing cost to reject routes likely leading to loops. To minimize the effort to integrate TARF and the existing protocol and to reduce the overhead, when an existing routing protocol does not provide any anti loop mechanism, we adopt the following mechanism to detect routing loops. To detect loops, the Trust Manager on N reuses the table of <node id of a source node, a forwarded sequence interval [a, b] with a significant length in last period. If N finds that a received data packet is already in that record table, not only will the packet be discarded, but the Trust Manager on N also degrades its next-hop node's trust level. If that next hop node is b, then T_0i_d Nb is the latest trust level value of b.

$$T_{new\_Nb} = \begin{cases} (1 - w_{degrade}) \times T_{old\_Nb} + w_{degrade} \times Loop, \\ \text{if } Loop = 0. \\ (1 - w_{upgrade}) \times T_{old\_Nb} + w_{upgrade} \times Loop, \\ \text{if } Loop = 1. \end{cases}$$

Once a loop has been detected by N for a few times so that the trust level of the next-hop node is too low, N will change its next-hop selection, thus that loop is broken. Though N cannot tell which node should be held responsible for the occurrence of a loop, degrading its next-hop node's trust level gradually leads to the breaking of the loop. On the other hand, to detect the traffic misdirection by nodes exploiting the replay of routing information, Trust Manager on N compares N's stored table of <node id of a source node, forwarded sequence interval [a, b] with a significant length> recorded in last period with the broadcast messages from the base station about data delivery. It computes the ratio of the number of successfully delivered packets which are forwarded by this node to the number of those forwarded data packets, denoted as Delivery Ratio. Then, N's Trust Manager updates its next-hop node b's trust level as follows:

$$T_{new\_Nb} = \begin{cases} (1 - w_{degrade}) \times T_{old\_Nb} \\ \; + w_{degrade} \times DeliveryRatio, \\ \text{if } DeliveryRatio < T_{old\_Nb}. \\ \\ (1 - w_{upgrade}) \times T_{old\_Nb} \\ \; + w_{upgrade} \times DeliveryRatio, \\ \text{if } DeliveryRatio >= T_{old\_Nb}. \end{cases}$$

5.5 Analysis on Energy Watcher and Trust Manager

Now that a node N relies on its Energy Watcher and Trust Manager to select an optimal neighbor as its next-hop node, we would like to clarify a few important points on the design of Energy Watcher and Trust Manager.

First, as described in Section 3.1, the energy cost report is the only information that a node is to passively receive and take as "fact." It appears that such acceptance of energy cost report could be a pitfall when an attacker or a compromised node forges false report of its energy cost. Note that the main interest of an attacker is to prevent data delivery rather than to trick a data packet into a less efficient route, considering the effort it takes to launch an attack. As far as an attack aiming at preventing data delivery is concerned, TARF well mitigates the effect of this pitfall through the operation of Trust Manager. Note that the Trust Manager on one node does not take any recommendation from the Trust Manager on another node. If an attacker forges false energy report to form a false route, such intention will be defeated by Trust Manager: when the Trust Manager on one node finds out the many identifies a Trust Manager on one node finds out the many delivery failures from the broadcast messages of the base station, it degrades the trust level of its current next-hop node; when that trust level goes below certain threshold, it causes the node to switch to a more promising next-hop node.

Second, Trust Manager identities the low trustworthiness of various attackers misdirecting the multihop routing, especially those exploiting the replay of routing information. It is noteworthy that Trust Manager does not distinguish whether an error or an attack occurs to the next-hop node or other succeeding nodes in the route. It seems unfair that Trust Manager downgrades the trust level of an honest next-hop node while the attack occurs somewhere after that next-hop node in the route. Contrary to that belief, Trust Manager significantly improves data delivery ratio in the existence of attack attempts of preventing data delivery. First of all, it is often difficult to identify an attacker who participates in the network using an id "stolen" from another legal node. For example, it is extremely difficult to detect a few attackers colluding to launch a combined wormhole and sinkhole attack [4]. Additionally, despite the certain inevitable unfairness involved, Trust Manager encourages a node to choose another route when its current route frequently fails to deliver data to the base station. Though only those legal neighboring nodes of an attacker might have correctly identified the adversary, our evaluation results indicate that the strategy of switching to a new route without identifying the attacker actually significantly improves the network performance, even with the existence of wormhole and sinkhole attacks. Figure 3 gives an example to illustrate this point. In this example, nodes A, B, C and D are all honest nodes and not compromised. Node A has node B as its current next-hop node while node B has an attacker node as its next-hop node. The attacker drops every packet received and thus any data packet passing node A will not arrive at the base station. After a while, node A discovers that the data packets it forwarded did not get delivered. The TrustManager on node A starts to degrade the trust level of its current next-hop node B although node B is absolutely honest. Once that trust level becomes too low, node A decides to select node C as its new next-hop node. In this way, node A identifies a better and successful route (A - C - D - base). In spite of the sacrifice of node B's trust level, the network performs better. Further, concerning the stability of routing path, once a valid node identifies a trustworthy honest neighbor as its next-hop node, it tends to keep that next-hop selection without considering other seemingly attractive nodes such as a fake base station. That tendency is caused by both the preference to maintain stable routes and the preference to highly trustable nodes.

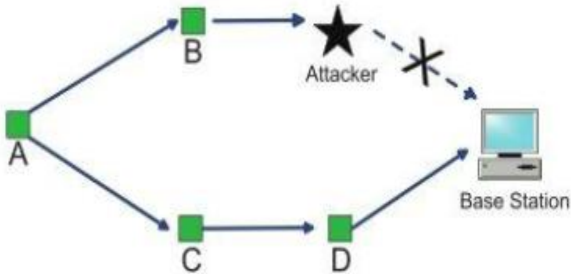


Figure 3: An Example to Illustrate how Trust Manager Works

Finally, we would like to stress that TARF is designed to guard a WSN against the attacks misdirecting the multihop routing, especially those based on identity theft through replaying the routing information. Other types of attacks such as the denial-of-service [3] attacks are out of the discussion of this paper. For instance, we do not address the attacks of injecting into the network a number of data packets containing false sensing data but authenticated (possibly through hacking). That type of attacks aim to exhaust the network resource instead of misdirecting the routing. However, if the attacker intends to periodically inject a few routing packets to cause wrong route, such attacks can still be defended by TARF through Trust Manager.

7. CONCLUSIONS

We have designed and implemented TARF, a robust trust-aware routing framework for WSNs, to secure multihop routing in dynamic WSNs against harmful attackers exploiting the replay of routing information. TARF focuses on trustworthiness and energy efficiency, which are vital to the survival of a WSN in a hostile environment. With the idea of trust management, TARF enables a node to keep track of the trustworthiness of its neighbors and thus to select a reliable route. Our main contributions are listed as follows:

- Unlike previous efforts at secure routing for WSNs, TARF effectively protects WSNs from severe attacks through replaying routing information; it requires neither tight time synchronization nor known geographic information.
- The resilience and scalability of TARF are proved through both extensive simulation and empirical evaluation

with large-scale WSNs; the evaluation involves both static and mobile settings, hostile network conditions, as well as strong attacks such as wormhole attacks and Sybil attacks.

- We have implemented a ready-to-use TinyOS module of TARF with low overhead; as demonstrated in the paper, this TARF module can be integrated into existing routing protocols with the least effort, thus producing secure and efficient fully functional protocols.
- Finally, we demonstrate a proof-of-concept mobile target detection application that is built on top of TARF and is resilient in the presence of an antidetection mechanism that indicates the potential of TARF in WSN applications.

REFERENCES

1. G. Zhan, W. Shi, and J. Deng, “Tarf: A Trust-Aware Routing Framework for Wireless Sensor Networks,” Proc. Seventh European Conf. Wireless Sensor Networks (EWSN ’10), 2010.
2. F. Zhao and L. Guibas, Wireless Sensor Networks: An Information Processing Approach. Morgan Kaufmann, 2004.
3. A. Wood and J. Stankovic, “Denial of Service in Sensor Net-works,” Computer, vol. 35, no. 10, pp. 54-62, Oct. 2002.
4. C. Karlof and D. Wagner, “Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures,” Proc. First IEEE Int’l Workshop Sensor Network Protocols and Applications, 2003.
5. M. Jain and H. Kandwal, “A Survey on Complex Wormhole Attack in Wireless Ad Hoc Networks,” Proc. Int’l Conf. Advances in Computing, Control, and Telecomm. Technologies (ACT ’09), pp. 555- 558, 2009.
6. A. Rezgui and M. Eltoweissy, “Tarp: A Trust-Aware Routing Protocol for Sensor-Actuator Networks,” Proc. IEEE Int’l Conf. Mobile Adhoc and Sensor Systems (MASS ’07), 2007.
7. A. Abbasi and M. Younis, “A Survey on Clustering Algorithms for Wireless Sensor Networks,” Computer Comm., vol. 30, pp. 2826-2841, Oct. 2007.
8. A. Perrig, R. Szewczyk, W. Wen, D. Culler, and J. Tygar, “SPINS: Security Protocols for Sensor Networks,” Wireless Networks J., vol. 8, no. 5, pp. 521-534, Sept. 2002.
9. R. Watro, D. Kong, S. Cuti, C. Gardiner, C. Lynn, and P. Kruus, “Tinypk: Securing Sensor Networks with Public Key Technology,” Proc. Second ACM Workshop Security of Ad Hoc and Sensor Networks (SASN ’04), pp. 59-64, 2004.
10. A. Liu and P. Ning, “Tinyecc: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks,” Proc. Seventh Int’l Conf. Information Processing in Sensor Networks (IPSN ’08), pp. 245-256, 2008.
11. I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “A Survey on Sensor Networks,” IEEE Comm. Magazine, vol. 40, no. 8, pp. 102-114, Aug. 2002.
12. H. Safa, H. Artail, and D. Tabet, “A Cluster-Based Trust-Aware Routing Protocol for Mobile Ad Hoc Networks,” Wireless Net-works, vol. 16, no. 4, pp. 969-984, 2010.

13. W. Xue, J. Aiguo, and W. Sheng, “Mobile Agent Based Moving Target Methods in Wireless Sensor Networks,” Proc. IEEE Int’l Symp. Comm. and Information Technology (ISCIT ’05), vol. 1, pp. 22-26, 2005.

14. J. Hee-Jin, N. Choon-Sung, J. Yi-Seok, and S. Dong-Ryeol, “A Mobile Agent Based Leach in Wireless Sensor Networks,” Proc. 10th Int’l Conf. Advanced Comm. Technology (ICACT ’08), vol. 1, pp. 75-78, 2008.

15. J. Al-Karaki and A. Kamal, “Routing Techniques in Wireless Sensor Networks: A Survey,” Wireless Comm., vol. 11, no. 6, pp. 6-28, Dec. 2004.

16. C. Karlof, N. Sastry, and D. Wagner, “Tinysec: A Link Layer Security Architecture for Wireless Sensor Networks,” Proc. ACM Int’l Conf. Embedded Networked Sensor Systems (SenSys ’04), Nov. 2004.

17. S. Chang, S. Shieh, W. Lin, and C. Hsieh, “An Efficient Broadcast Authentication Scheme in Wireless Sensor Networks,” Proc. ACM Symp. Information, Computer and Comm. Security (ASIACCS ’06), pp. 311-320, 2006.

18. K. Ren, W. Lou, K. Zeng, and P. Moran, “On Broadcast Authentication in Wireless Sensor Networks,” IEEE Trans. Wireless Comm., vol. 6, no. 11, pp. 4136-4144, Nov. 2007.

19. P. De, Y. Liu, and S.K. Das, “Modeling Node Compromise Spread in Wireless Sensor Networks Using Epidemic Theory,” Proc. Int’l Symp. World of Wireless, Mobile and Multimedia Networks (WoWMoM ’06), pp. 237-243, 2006.

20. A. Woo, T. Tong, and D. Culler, “Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks,” Proc. First ACM Int’l Conf. Embedded Networked Sensor Systems (SenSys ’03), Nov. 2003.

21. S. Ganeriwal, L. Balzano, and M. Srivastava, “Reputation-Based Framework for High Integrity Sensor Networks,” ACM Trans. Sensor Networks, vol. 4, pp. 1-37 2008.

22. G. Zhan, W. Shi, and J. Deng, “Poster Abstract: Sensortrust—A Resilient Trust Model for WSNs,” Proc. Seventh Int’l Conf. Embedded Networked Sensor Systems (SenSys ’09), 2009.

23. C. Perkins and P. Bhagwat, “Highly Dynamic Destination- Sequenced Distance-Vector Routing (dsdv) for Mobile Compu-ters,” ACM SIGCOMM Computer Comm. Rev., vol. 24, no. 4, pp. 234-244, 1994.