International Journal of Engineering Research and General Science Volume 4, Issue 6,November-December, 2016
ISSN 2091-2730

Trust Aware Routing Framework

Mr.C.M Jadhav , Miss Ummeaiyman Shaikh

BIGCE, Solapur University, uamshaikh@gmail.com

Abstract- The multi-hop routing in wireless sensor networks (WSNs) offers little protection against identity deception through replaying routing information. An adversary can exploit this defect to launch various harmful or even devastating attacks against the routing protocols, including *sinkhole* attacks, wormhole attacks and *Sybil* attacks. Traditional cryptographic techniques or efforts at developing trust-aware routing protocols do not effectively address this severe problem. To secure the WSNs against adversaries misdirecting the multi-hop routing, we have design and implemented TARF, a robust trust-aware routing framework for dynamic WSNs.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.

***Index Terms*- :** CTP – Collection Tree Routing Protocol, EWMA-exponentially weighted moving average,RPGM-Reference Point Group Mobility TARF-Trust Aware Routing Framework, WSN – Wireless Sensor Network

I. INTRODUCTION

Wireless sensor networks (WSNs) mainly supports military applications and forest fire monitoring. A WSN comprises battery-powered sensor nodes with extremely limited processing capabilities. With a narrow radio communication range, a sensor node wirelessly sends Wireless sensor networks (WSNs) . With a narrow radio communication range, a sensor node wirelessly sends messages to a base station via a multi-hop path. However, the multi-hop routing of WSNs often becomes the target of malicious attacks. An attacker may tamper nodes physically, create traffic collision with seemingly valid transmission, drop or misdirect messages in routes, or jam the communication channel by creating radio interference. This paper focuses on the kind of attacks in which adversaries misdirect network traffic by identity deception through replaying routing information. Based on identity deception, the adversary is capable of launching harmful and hard-to-detect attacks against routing such as selective forwarding, wormhole attacks, sinkhole attacks and Sybil attacks.

II. DESIGN CONSIDERATIONS

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.

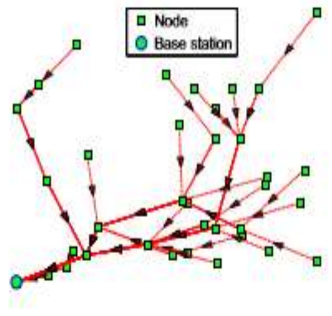


Fig.1.Multi-hop routing for data collection of a WSN

AUTHENTICATION REQUIREMENTS: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 A BROADCAST MESSAGE FROM THE BASE STATION.TARF USES TRUSTMANAGER.

Goals : TARF mainly guards a WSN against the attacks misdirecting the multi-hop routing, especially those based on identity theft through replaying the routing information. 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.Here,*throughput* at a moment is computed over the period from the beginning time (0) until that particular moment. Note that single-hop re-transmission may happen, and that duplicate packets are considered as one packet as far as *throughput* is concerned. *Through- put* reflects how efficiently the network is collecting and delivering data.T*hroughput should be high*.
***Energy Efficiency**:* 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 re-transmission should be given enough attention when considering energy cost since each re-transmission 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. Here,the energy consumption depends on the number of hops, i.e. the number of one-hop transmissions occurring. It is abbreviated as *hop-per-delivery*.
Scalability & Adaptability :It should support large magnitude and high dynamic data. We will evaluate the scalability and adaptability of TARF through experiments with large-scale WSNs and under mobile and hash network conditions.

III. DESIGN OF TARF

Before introducing the detailed design, we first introduce several necessary notion 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.
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 EMPLOYED BY TARF. A BROADCAST MESSAGE FROM THE BASE STATION IS ﬂOODED TO THE WHOLE NETWORK.
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 acknowledgement. A broadcast message from the base station is flooded .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.

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, *EnergyWatcher* and *TrustManager*, run on the node (Figure 2). *EnergyWatcher* 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. *TrustManager* 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 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 *EnergyWatcher*.

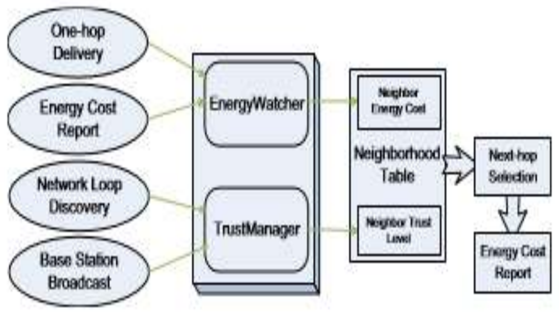


Fig. 2 : Design of TARF

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.. During each period, the *EnergyWatcher* 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 *TrustManager* 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.

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 (Section 5.1) and proved effective (Section 5.3, 5.4). 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 {109, 110, 111, 150, 151} in last period. Then [112, 149] is an undelivered sequence interval; [109, 151] 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 *TrustManager* 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 afterwards. 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 *EnergyWatcher* (see Section 3.3). 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 *TrustManager* on the neighbors of the base station. Therefore, 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.

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.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 trade-off between TNb and E_{Nb} , with ENb and TNb being b's energy cost and trust level value in the neighborhood table respectively. Basically, E_{Nb} reflects the energy cost of delivering a packet to the base station from N assuming that all the nodes in the route are honest; $1 / T_{Nb}$ approximately reflects the number of the needed attempts to send a packet from N to the base station via multiple hops before such an attempt succeeds, considering the trust level of b.

Thus, E_{Nb} / T_{Nb} combines the trustworthiness and energy cost. However, the metric E_{Nb} / T_{Nb} suffers from the fact that an adversary may falsely reports extremely low energy cost to attract traffic and thus resulting in a low value of E_{Nb} / T_{Nb} even with a low TNb. Therefore, TARF prefers nodes with significantly higher trust values;. For deciding the next-hop node, a specific trade-off between T_{Nb} and $E_{Nb} T_{Nb}$ is demonstrated in Figure 5 (see Section 5.2).

EnergyWatcher : Here we describe how a node N's EnergyWatcher computes the energy cost ENb for its neighbor b in N's neighborhood table and how N decides its own energy cost E_N. 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 re-transmission may occur until the acknowledgement is received or the number of re-transmissions reaches a certain threshold. The cost caused by one-hop retransmissions should be included when computing E_{Nb}. Suppose N decides that A should be its next-hop node after comparing energy cost and trust level.

Then N's energy cost is $E_N = _{ENA.}$ Denote EN→b as the average energy cost of successfully delivering a data packet from N to its neighbor b with one hop. Note that the re- transmission cost needs to be considered. With the above notations, it is straightforward to establish the following relation: $E_{Nb} = E_N \rightarrow b + E_b$

Since each known neighbor b of N is supposed to broadcast its own energy cost Eb to N, to compute E_{Nb}, N still needs to know the value EN→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 acknowledgement is received as follows:

∞ X i= 1 Psucc · (1 − Psucc)$^{i=1}$ =1 / Psucc

Denote Eunit as the energy cost for node N to send a unit-sized data packet once regardless of whether it is received or not. Then we have ENb =Eunit/ Psucc+ Eb

The remaining job for computing ENb is to get the probability Psucc 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 Psucc. Instead, after each transmission from N to b, N's EnergyWatcher will update Psucc 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 acknowledgement is received; otherwise, 0. Given Ack and the last probability value of an acknowledged transmission $P_{old\ succ}$, an intuitive way is to use a simply weighted average of Ack and $P_{oldsucc}$ as the value of Pnew succ. That is what is essentially adopted in the aging mechanism.However, that method used against sleeper attacks still suffers periodic attacks. To solve this problem, we update the Psucc value using two different weights as in our previous work , a relatively big $W_{degrade} \in (0,1)$ and a relatively small $W_{upgrade} \in (0,1)$ as follows:

pnew succ = $(1 - W_{degrade}) \times$ $_{\text{Pold succ}}$ $+ W_{degrade} \times$ Ack, if Ack = 0.

$(1 - W_{upgrade}) \times$ $_{\text{Pold succ}}$ $+ W_{upgrade} \times$ Ack, if Ack =1 .

The two parameters $W_{degrade}$ and $W_{upgrade}$ allow flexible application requirements. $W_{degrade}$ and $W_{upgrade}$ 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, $W_{degrade}$ 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 require.0es many more positive transactions, then $W_{upgrade}$ should be assigned a relatively low value.

TrustManager : A node N's TrustManager 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, TNb 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. In that case, when integrating TARF into those protocols with anti-loop mechanisms, TrustManager may solely depend on the broadcast from the base station to decide the trust level; we adopted such a policy when implementing TARF later (see Section 5). If anti-loop mechanisms are both enforced in the TARF component and the routing protocol that integrates TARF, then the resulting hybrid protocol may overly react towards 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 [32]. 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 TrustManager on N reuses the table of <node id of a source node, a forwarded sequence interval [a, b] with a significant length> (see Section 3.2) 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 TrustManager on N also degrades its next-hop node's trust level. If that next-hop node is b, then Told Nb is the latest trust level value of b. We use a binary variable Loop to record the result of loop discovery: 0 if a loop is received; 1 otherwise. As in the update of energy cost, the new trust level of b is

$T_{new\ Nb} =(1 - W_{degrade}) \times T_{old\ Nb} + W_{degrade} \times \text{Loop}$, if Loop = 0.

$= (1 - W_{upgrade}) \times T_{old\ Nb} + W_{upgrade} \times \text{Loop}$, if Loop = 1.

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 can not 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, TrustManager on 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 DeliveryRatio. Then N's TrustManager updates its next-hop node b's trust level as follows:

Tnew Nb =

$(1 - W_{degrade}) \times T_{old\ Nb} + W_{degrade} \times \text{DeliveryRatio}$, if DeliveryRatio < $T_{old\ Nb.}$

$(1 - W_{upgrade}) \times T_{old\ Nb} + W_{upgrade} \times \text{DeliveryRatio}$, if DeliveryRatio >= $T_{old\ Nb}$.

Analysis on EnergyWatcher and TrustManager : Now that a node N relies on its EnergyWatcher and TrustManager to select an optimal neighbor as its next- hop node, we would like to clarify a few important points on the design of EnergyWatcher and TrustManager. 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 TrustManager. Note that the TrustManager on one node does not take any recommendation from the TrustManager on another node. If an attacker forges false energy report to form a false route, such intention will be defeated by TrustManager: when the TrustManager 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.. 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 . Additionally, despite the certain inevitable unfairness involved, TrustManager 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. Fig 3 gives an example to illustrate this point. In this example, node 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.

0

Fig.3. An example to illustrate how TrustManager works.

Finally, we would like to stress that TARF is designed to guard a WSN against the attacks misdirecting the multi-hop routing, especially those based on identity theft through replaying the routing information.

IV. **Simulation**

In our experiments, initially, 35 nodes are randomly distributed within a 300*300 rectangular area, with unreliable wireless transmission. All the nodes have the same power level and the same maximal transmission range of 100m. Each node samples 6 times in every period; the timing gap between every two consecutive samplings of the same node is equivalent. We simulate the sensor network in 1440 consecutive periods. Regarding the network topology, we set up three types of network topologies. The first type is the static-location case under which all nodes stand still. The second type is a customized group-motion-with-noise case based on Reference Point Group Mobility (RPGM) model that mimics the behavior of a set of nodes moving in one or more groups . The last type of dynamic network incorporated in the experiments is the addition of scattered RF-shielded areas to the afore mentioned group-motion-with-noise case.

The performance of TARF is compared to that of a link connectivity-based routing protocol. With the Link-connectivity protocol, each node selects its next-hop node among its neighborhood table according to an link estimator based on exponentially weighted moving average (EWMA). The simulation results show, in the presence of misbehaviors, the throughput in TARF is often much higher than that in Link-connectivity; the hop-per- delivery in the Link-connectivity protocol is generally at least comparable to that in TARF.Both protocols are evaluated under three common types of attacks: (1) a certain node forges the identity of the based station by replaying broadcast messages, also known as the sinkhole attack; (2) a set of nodes colludes to form a forwarding loop; and (3) a set of nodes drops received data packets. Generally, under these common attacks, TARF produces a substantial improvement over Link-connectivity in terms of data collection and energy efficiency. Further, we have evaluated TARF under more severe attacks: multiple moving fake bases and multiple Sybil attackers. TARF succeeds in achieving a steady improvement over the Link-connectivity protocol.

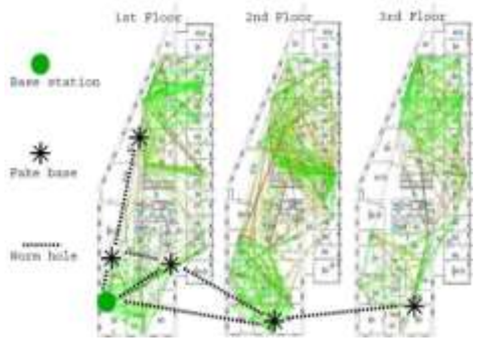
Incorporation of TARF into Existing Protocols : To demonstrate how this TARF implementation can be integrated into the existing protocols with the least effort, we incorporated TARF into a collection tree routing protocol (CTP). The CTP protocol is efficient, robust, and reliable in a network with highly dynamic link topology. It quantifies link quality estimation in order to choose a next-hop node. The software platform is TinyOS 2.x.

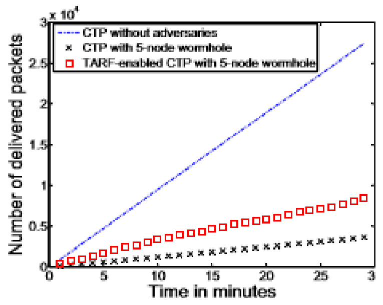
To perform the integration, after proper interface wiring, invoke the TrustControl.start command to enable the trust evaluation; call the Record.addForwarded command for a non-root node to add forwarded record once a data packet has been forwarded; call the Record.addDelivered command for a root to add delivered record once a data packet has been received by the root. Finally, inside the CTP's task to update the routing path, call the Record.getTrust command to retrieve the trust level of each next-hop candidate; an algorithm taking trust into routing consideration is executed to decide the new next-hop neighbor.(See Figure 5).

Similar to the original CTP's implementation, the implementation of this new protocol decides the next-hop neighbor for a node with two steps (see Figure 5): Step 1 traverses the neighborhood table for an optimal candidate for the next hop; Step 2 decides whether to switch from the current next-hop node to the optimal candidate found. For Step 1, as in the CTP implementation, a node would not consider those links congested, likely to cause a loop, or having a poor quality lower than a certain threshold. This new implementation prefers those candidates with higher trust levels; in certain circumstances, regardless of the link quality, the rules deems a neighbor with a much higher trust level to be a better candidate (see Figure 5). The preference of highly trustable candidates is based on the following consideration: on the one hand, it creates the least chance for an adversary to misguide other nodes into a wrong routing path by forging the identity of an attractive node such as a root; on the other hand, forwarding data packets to a candidate with a low trust level would result in many unsuccessful link-level transmission attempts, thus leading to much re-transmission and a potential waste of energy. When the network throughput becomes low and a node has a list of low-trust neighbors, the node will exclusively use the trust as the criterion to evaluate those neighbors for routing decisions. As show in Figure 5, it uses trust/cost as a criteria only when the candidate has a trust level above certain threshold. The reason is, the sole trust/cost criteria could be exploited by an adversary replaying the routing information from a base station and thus pretending to be an extremely attractive node. As for Step 2, compared to the CTP implementation, we add two more circumstances when a node decides to switch to the optimal candidate found at Step 1: that candidate has a higher trust level, or the current next-hop neighbor has a too low trust level.

V. EmpiricalEvaluationonMotelab

We evaluated the performance of TARF against a combined sinkhole and wormhole attack on Motelab at Harvard University. 184 TMote Sky sensor motes were deployed across many rooms at three floors in the department building (see Figure 6), with two to four motes in most rooms. Around 97 nodes functioned properly while the rest were either removed or disabled. Each mote has a 2.4GHz Chipcon CC2420 radio with an indoor range of approximately 100 meters. In Figure 6, the thin green lines indicate the direct (one-hop) wireless connection between motes. Certain wireless connection also exists between nodes from different floors.We developed a simple data collection application in TinyOS 2.x that sends a data packet every five seconds to a base station node (root) via multi-hop. This application was executed on 91 functioning non-root nodes on Mote- lab. For comparison, we used CTP and the TARF-enabled CTP implementation as the routing protocols for the data collection program separately. The TARF-enabled CTP has a TARF period of 30 seconds. We conducted an attack with five fake base stations that formed a wormhole. As in Figure 6, whenever the base station sent out any packet, three fake base stations which overheard that packet replayed the complete packet without changing any content including the node id. Other fake base stations overhearing that replayed packet would also replay the same packet. Each fake base station essentially launched a sinkhole attack. Note that there is a distinction between such malicious replay and the forwarding when a well-behaved node receives a broadcast from the base station. When a well-behaved node forwards a broadcast packet from the base station, it will include its own id in the packet so that its receivers will not recognize the forwarder as a base station. We conducted the first experiment by uploading the program with the CTP protocol onto 91 motes (not including those 5 selected motes as fake bases in later experiments), and no attack was involved here





(a) All three floors.

Fig. 6. Connectivity map of Motelab   Fig.7 . Empirical comparison of CTP and TARF- enabled CTP on Motelab

Then, in another experiment, in addition to programming those 91 motes with CTP, we also programmed the five fake base stations so that they stole the id the base station through replaying. In the last experiment, we programmed those 91 motes with the TARF-enabled CTP, and programmed the five fake base stations as in the second experiment.

Each of our programs run for 30 minutes. As illustrated in Figure 7(a), the existence of the five wormhole attackers greatly degraded the performance of CTP: the number of the delivered data packets in the case of CTP with the five-node wormhole is no more than 14% that in the case of CTP without adversaries. The TARF-enabled CTP succeeded in bringing an immense improvement over CTP in the presence of the five-node wormhole, almost doubling the throughput. That improvement did not show any sign of slowing down as time elapsed. The number of nodes from each floor that delivered at least one data packet in each six-minute sub-period is plotted in Figure 7. On each floor, without any adversary, at least 24 CTP nodes were able to find a successful route in each six minute. However, with the five fake base stations in the wormhole, the number of CTP nodes that could find a successful route goes down to 9 for the first floor; it decreases to no more than 4 for the second floor; as the worst impact, none of the nodes on the third floor ever found a successful route. A further look at the data showed that all the nine nodes from the first floor with successful delivery record were all close to the real base station. The CTP nodes relatively far away from the base station, such as those on the second and the third floor, had little luck in making good routing decisions. When TARF was enabled on each node, most nodes made correct routing decisions circumventing the attackers. That improvement can be verified by the fact that the number of the TARF- enabled nodes with successful delivery record under the threat of the wormhole is close to that of CTP nodes with no attackers, as shown in Figure 7.

VI. **CONCLUSIONS**

We have designed and implemented TARF, a robust trust-aware routing framework for WSNs, to secure multi-hop 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. (1) 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. (2) The resilience and scalability of TARF is 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.
(3) 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.

REFERENCES:

[1] G. Zhan, W. Shi, and J. Deng, "Tarf: A trust-aware routing framework for wireless sensor networks," in *Proceeding of the 7th European Conference on Wireless Sensor Networks (EWSN'10)*, 2010.

[2] F. Zhao and L. Guibas, *Wireless Sensor Networks: An Information Processing Approach*. Morgan Kaufmann Publishers, 2004.

[3] A. Wood and J. Stankovic, "Denial of service in sensor networks,"

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,” in *Proceedings of the 1st IEEE International Workshop on Sensor Network Protocols and Applications*, 2003.

[5] M. Jain and H. Kandwal, “A survey on complex wormhole attack in wireless ad hoc networks,” in *Proceedings of International Con- ference on Advances in Computing, Control, and Telecommunication Technologies (ACT ’09)*, 28-29 2009, pp. 555 –558.

[6] I. Krontiris, T. Giannetsos, and T. Dimitriou, “Launching a sink- hole attack in wireless sensor networks; the intruder side,” in *Proceedings of IEEE International Conference on Wireless and Mobile Computing, Networking and Communications(WIMOB ’08)*, 12-14

2008, pp. 526 –531.

[7] J. Newsome, E. Shi, D. Song, and A. Perrig, “The sybil attack in sensor networks: Analysis and defenses,” in *Proc. of the 3rd International Conference on Information Processing in Sensor Networks (IPSN’04)*, Apr. 2004.

[8] L. Bai, F. Ferrese, K. Ploskina, and S. Biswas, “Performance analy- sis of mobile agent-based wireless sensor network,” in *Proceedings of the 8th International Conference on Reliability, Maintainability and Safety (ICRMS 2009)*, 20-24 2009, pp. 16 –19..

[9] L. Zhang, Q. Wang, and X. Shu, “A mobile-agent-based middleware for wireless sensor networks data fusion,” in *Proceedings of Instrumentation and Measurement Technology Conference (I2MTC’09)*, 5-7 2009, pp. 378 –383.

[10] W. Xue, J. Aiguo, and W. Sheng, “Mobile agent based moving target methods in wireless sensor networks,” in *IEEE International Symposium on Communications and Information Technology (ISCIT2005)*, vol. 1, 12-14 2005, pp. 22 – 26.

[11] J. Hee-Jin, N. Choon-Sung, J. Yi-Seok, and S. Dong-Ryeol, “A mobile agent based leach in wireless sensor networks,” in *Proceedings of the 10th International Conference on Advanced Communication Technology (ICACT 2008)*, vol. 1, 17-20 2008, pp. 75 –78.

[12] J. Al-Karaki and A. Kamal, “Routing techniques in wireless sensor networks: a survey,” *Wireless Communications*, vol. 11, no. 6, pp.

6–28, Dec. 2004.

[13] C. Karlof, N. Sastry, and D.Wagner, “Tinysec: A link layer security architecture for wireless sensor networks,” in *Proc. of ACM SenSys2004*, Nov. 2004.

[14] A. Perrig, R. Szewczyk, W. Wen, D. Culler, and J. Tygar, “SPINS: Security protocols for sensor networks,” *Wireless Networks Journal*

(WINET), vol. 8, no. 5, pp. 521–534, Sep. 2002.

[15] R. Watro, D. Kong, S. Cuti, C. Gardiner, C. Lynn, and P. Kruus, “Tinypk: securing sensor networks with public key technology,”

in *Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks (SASN ’04)*. New York, NY, USA: ACM, 2004, pp. 59–64

IJERGS