Volume 5 Issue I, January 2017
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

A Trust Directing System for Wireless Sensor Networks

SK. Shahida[1], SK. Shahina[2]
[1,2]Assistant Professor, Department of CSE, Tirumala Engineering College, JNTU-Kakinada, AP, INDIA,522601

Abstract: ***Mobile nodes in military environments such as a battlefield or a hostile region are likely to suffer from intermittent network connectivity and frequent partitions. Disruption-tolerant network (DTN) technologies are becoming successful solutions that allow wireless devices carried by soldiers to communicate with each other and access the confidential information or command reliably by exploiting external storage nodes. Some of the most challenging issues in this scenario are the enforcement of authorization policies and the policies update for secure data retrieval. Cipher text policy attribute-based encryption (CP-ABE) is a promising cryptographic solution to the access control issues. However, the problem of applying CP-ABE in decentralized DTNs introduces several security and privacy challenges with regard to the attribute revocation, key escrow, and coordination of attributes issued from different authorities. In this paper, we propose a secure data retrieval scheme using CP-ABE for decentralized DTNs where multiple key authorities manage their attributes independently. We demonstrate how to apply the proposed mechanism to securely and efficiently manage the confidential data distributed in the disruption-tolerant military network***

Keywords: Access control, attribute-based encryption (ABE), disruption-tolerant network (DTN), multiauthority, secure data retrieval

I. INTRODUCTION

Wireless sensor networks (WSNs) are ideal candidates for applications, for example, military surveillance and forest fire monitoring to report recognized occasions of interest. With a narrow radio correspondence run, a sensor node wirelessly sends messages to a base station by means of a multi-hop path. In any case, the multi-hop routing of WSNs regularly turns into the objective of malicious attacks. In such an attack, the attacker may tamper nodes physically, make traffic collision with apparently legitimate transmission, drop or mislead messages in routes or jam the communication channel by making radio interference [18]. This paper concentrates on the kind of attack in which an adversary misleads packets by identity deception through replaying routing data. With such identity deception, the adversary is capable of launching harmful and hard-to-detect attacks to misdirect traffic, such as selective forwarding as well as wormhole and sinkhole attacks [8].

As a successful and easy to-implement type of attack, a malicious node simply re-plays all the routing data sent from another substantial node to fashion the last node's identity, in this way misleading the network traffic. Those packets, including their original headers, are replayed with no 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 first legitimate node, which is known as a wormhole attack. Since a node in a WSN for the most part depends exclusively on the bundles got to think about the sender's identity, replaying routing packets permits 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. In a selective forwarding attack, 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 over-hearing techniques [8]. 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. Such a fake base station could lure more than half the traffic, creating a "black hole".

Unfortunately, most existing routing protocols for WSNs either concentrate on energy efficiency [1] assuming that every node is straightforward with its identity, or they attempt to exclude unapproved support by encrypting data and authenticating packets. As a matter of fact, it is vital to consider energy usage for battery-powered sensor nodes and the robustness of routing under topological changes and common faults in a wild environment. Be that as it may, it is likewise critical to join security as a standout amongst the most essential objectives; in the mean time, even with flawless encryption and authentication, by replaying routing data, a malicious

Volume 5 Issue I, January 2017
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

node can in any case take an interest in the network utilizing another valid node's identity. These studies target general specially appointed systems and distributed systems yet not asset compelled WSNs. Also, they don't address attacks emerging from the replay of routing information.

Now, to battle against the "data fraud" danger emerging from bundle replaying, we bring trust administration into WSNs, proposing A trust directing system for wireless sensor networks.

II. ASSUMPTIONS AND GOALS

We target secure directing for information gathering, which are one of the most essential elements of WSNs. In an information accumulation, sensor nodes
end sampled data to a

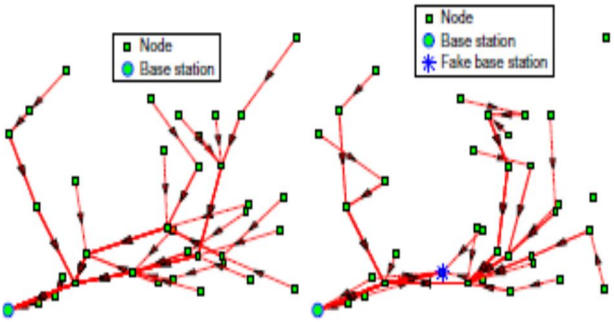

Fig: Multihop routing:(a)Normal Scenarios (b)A fake base station attracts traffic

remote base station with the guide of middle nodes, as in Figure 1(a). It is feasible for an enemy to replay every one of the packets from a base station and therefore to produce the identity of the base station. Such deception could bring about the accompanying circumstance: a lot of packets are pulled in to this fake base station and are never delivered to the genuine base station (see Figure

In spite of the fact that there could be more than one base station, our routing methodology is not influenced by the quantity of base stations; to streamline our discussion, we will accept that there is just a single base station. Further, we expect no information aggregation is included. Regardless, our approach can at present be connected to static-cluster-based WSNs, where information is amassed by static groups before being transferred. In a static-cluster-based WSN, cluster headers themselves shape a sub arrange; after specific information achieve a group header, the collected information will be steered to a base station just through such a sub-organize comprising of cluster headers. Our system can then be connected to this sub-system to accomplish secure directing for static-cluster-based WSNs.

Furthermore, we make certain presumptions with respect to the organization of packets in A Trust Directing System for Wireless sensor networks, we expect all data packets and routing packets, including their packet headers, are validated; a packet can be sent simply after its validness is checked. Whether data encryption is actualized can be chosen by the application. Each data packet is expected to have in any event the accompanying fields: the sender id, the sender sequence number, the next-hop id (the receiver in this one-hop transmission), the source id (the node that starts the data), and the source's sequence number. We demand that the source node's data ought to be incorporated for the accompanying reasons. Initially, that permits the base station to recognize which data packets are started however undelivered; Second, a WSN can't manage the cost of the overhead to transmit all the one-hop data to the base station. With respect to routing packets, they ought to have in any event the accompanying fields: the source id, the source's sequence number, and the following next-hop id. In addition, we assume that after receiving a data packet, a node will send out an acknowledgement packet.

High Throughput: *Throughput* is defined as the ratio of the number of data packets delivered to the base station to the number of all sampled data packets. Take note that single-hop re-transmission may happen, and that indistinguishable packets more than once transmitted are considered as one packet as far as throughput is concerned. Rather than a particular information, clients for the most part think significantly more about throughput. Here we see high throughput as one of our most imperative objectives.

Energy Efficiency: Efficient energy utilization is noteworthy for battery-fueled sensor nodes, and information transmission represents a noteworthy part of energy utilization. We assess energy effectiveness by the average energy cost to effectively convey unit-sized information from a source node to the base station. Take note of that link-level re-transmission ought to be sufficiently given consideration while considering energy cost since every re-transmission causes a discernible increment in energy utilization.

Volume 5 Issue I, January 2017
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

On the off chance that each node in a WSN devours around a similar energy to transmit a unit-sized information parcel, we can utilize another metric hop-per-delivery to assess energy efficiency. Under that supposition, the energy consumption relies on upon the number of hops, i.e. the number of one-hop transmissions happening. To assess how efficiently energy is utilized, we can measure the average hops per delivery, i.e., the quantity of all hops partitioned by the quantity of all delivered data packets, shortened as hop-per-delivery.

III. DESIGN OF A TRUST DIRECTING SYSTEM FOR WIRELESS SENSOR NETWORKS

A Trust Directing System for remote sensor networks secures the multi-hop routing in WSNs against intruders misusing the replay of routing data by assessing the reliability of neighboring nodes. It recognizes such intruders that mislead detectable system activity by their low dependability and courses information through ways going around those interlopers to accomplish agreeable throughput. A Trust Directing System for remote sensor systems is likewise energy-efficient, exceedingly scalable, and well adaptable.

Neighbor: For a node N, a neighbor (neighboring node) of N is a node that is reachable from N with one-hop remote transmission.

Trust level: For a node N, the trust level of a neighbor is a decimal number in [0, 1], speaking to N's assessment of that neighbor's level of dependability. In particular, the trust level of the neighbor is N's estimation of the likelihood that this neighbor accurately conveys information got 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 effectively convey a unit-sized data packet with this neighbor as its next-hop node, from N to the base station. That energy cost is signified as E in this paper.

A. Overview

A Trust Directing System for remote sensor networks incorporates dependability and energy effectiveness in settling on routing choices. For a node N to highway an information packet to the base station, N just needs to choose to which neighboring node it ought to forward the data packet. Once the information packet is sent to that next-hop node, the rest of the undertaking to convey the information to the base station is completely designated to it, and N is absolutely unconscious of what steering choice its next-hop node makes. To pick its next-bounce hub, N considers both the dependability and the energy efficiency of its neighbors. For that, N keeps up an area table with trust level qualities and energy cost values for certain known neighbors.

In A Trust Directing System for remote Sensor networks, notwithstanding data packet transmission, there are two sorts of routing in-arrangement that should be traded: communicate messages from the base station about undelivered information packets and energy cost report messages from every node. A communicate message from the base station is communicate to the entire system; every node getting a fresh communicate message from the base station will communicate it to every one of its neighbors once. The freshness of a communicate message is guaranteed by source succession number. The other kind of traded directing data is the energy cost report message from every node, which is communicate to just its neighbors once. Moreover, any node getting such a energy cost report message won't forward it.

For every node N in a WSN, to keep up such an area table with trust level qualities and vitality cost values for certain known neighbors, two parts, Energy-Watcher and Trust Manager, keep running on the hub (Figure 2). Vitality Watcher is in charge of recording the vitality cost for each known neighbor, in light of N's perception of one-bounce transmission to achieve its neighbors and the vitality cost report from those neighbors. Trust Manager is in charge of following trust level estimations of neighbors in view of system circle revelation and communicates messages from the base station about undelivered information parcels. When N can choose its next-hop neighbor as indicated by its neighbor-hood table, it conveys its vitality report message: it communicates to every one of its neighbors its energy cost to convey a bundle from the hub to the base station. The energy cost is registered as in Section 3.3 by Energy Watcher. Such a energy cost report likewise serves as the contribution of its collectors' Energy Watcher.

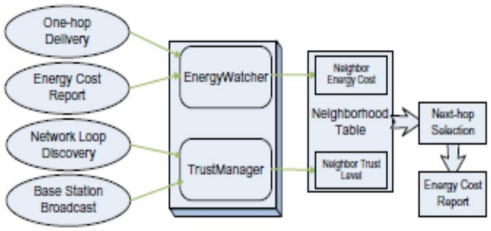

Fig 2:. Every node chooses a next-hop node in view of its neighborhood table, and communicates its energy cost inside its

neighborhood. To keep up this area table, Energy Watcher and Trust Manager on the node monitor related occasions (on the left) to record the vitality cost and the trust level estimations of its neighbors

B. Routing procedure

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 the information about un-delivered data packets during the past few periods to the whole network once, which triggers the exchange of routing information 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. In this way, no 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 undelivered data 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 from the base station. If a node does not change its next-hop node selection until the next broadcast 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 only when its current next-hop is not responding correctly. Next, we introduce the structure and exchange of routing information as well as how nodes make routing decisions in A Trust Directing System for wireless sensor networks

Structure and Exchange of Routing Information: A broadcast message from the base station fits into a fixed number of packets; in our implementation, it fits into one byte. Such a message consists of a few pairs of <the node id of a source node, an un-delivered sequence interval [a, b] with a significant length>. To reduce overhead, only a few such pairs are selected to be broadcast. The undelivered sequence interval [a, b] is explained as follows: the base station searches the source sequence numbers received in the past few periods, 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 the past two periods. Then [112, 149] is an undelivered sequence interval. 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 the entire source sequence numbers and finding undelivered sequence intervals. The reason for searching over more than one period is to identify as many undelivered data packets as possible. To illustrate that, consider this example: suppose the source sequence numbers of delivered data packets from node 2 are *{*1, 2, 3*}* for the 1st period and *{*200, 201, 203*}* for the 2nd period; then simply searching over a single period would not discover the un-delivered packets unless every node is required to send a fixed number of data packets over each period.

Accordingly, each node in the network stores a table of <the node id of a source node, a forwarded sequence interval [a, b] with a significant length> in the past few periods. 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 with undelivered sequence intervals, 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 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 *Energy Watcher* (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 *Trust Manager* 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 A Trust Directing System for wireless sensor networks decides routes in a WSN. Each

Volume 5 Issue I, January 2017
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

node *N* relies on its neighborhood table to select an optimal route, considering both energy consumption and reliability. A Trust Directing System for wireless sensor networks 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.Among the remaining known neighbors, *N* will select as its next-hop node a neighbor *b* with the minimal $^{E}_{Nb}$ $^{T}_{Nb}$ with E_{Nb} and T_{Nb} being *b*'s energy cost and trust level value in the neighborhood table respectively (see Section 3.3, 3.4). 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, comparing the values among *N*'s neighbors identifies a candidate with a minimal combined cost of energy and trustworthiness.

The remaining delivery task is fully delegated to that selected next-hop neighbor, and *N* is totally unaware of what routing decision its chosen neighbor is going to make. Next, the chosen node will repeat what *N* has done, i.e., delegating the left routing task to its own chosen next-hop neighbor. In this way, instead of finding out a complete path to the base station, each node is only responsible for choosing its next-hop node,

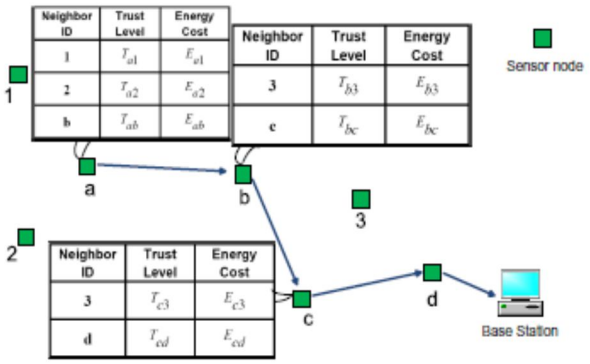

Fig:3. Routing illustration

Thus saving considerable cost in computation and routing information exchange. As an example shown in Figure 3, node *a* is trying to forward a packet to the base station. After comparing both the trust level and energy cost among its neighbors 1, 2 and *b*, *a* decides that *b* is the most promising next-hop node for data delivery and forwards the data packet to *b* immediately. *b* is free to make its own decision for routing the packet to the base station. *b* decides that its neighbor *c* is a better candidate than its neighbor 3. After that, the task is delegated to *c*, and *c* continues to delegate the job to *d*. Finally, *d* delivers the packet to the base station. 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. If we further assume that the one-hop transmission power of a unit-sized packet is the same for each node, the selected route will be the classical shortest path.

C. Energy Watcher

Here we describe how a node *N*'s *Energy Watcher* computes the energy cost E_{Nb} 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 re-transmissions 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 $E_N = E_{NA}$. Denote $E_{N\rightarrow b}$ 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 E_b to *N*, to compute E_{Nb}, *N* still needs to know the value $E_{N\rightarrow b}$, 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 sending needed before the acknowledgement is received as follows:

$$\sum_{i=1}^{\infty} i.\, psucc\,.(1 - psucc)^{i-1} = \frac{1}{psucc}$$

Denote E_{unit} 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

$$E_{Nb} = \frac{E_{unit}}{p_{succ}} + E_b$$

The remaining job for computing E_{Nb} 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 acknowledgement is received; otherwise, 0. Given *Ack* and the last probability value of an acknowledged transmission $p_{old\ succ}$, It uses a weighted average of *Ack* and $p_{oldsucc}$ as the new probability value $p_{new\ succ}$:

$p_{newsucc} = (1 - w) \times p_{oldsucc} + w \times Ack, w \in (0, 1)$,Where w can be chosen by specific protocols.

D. 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 undelivered data packets. For each neighbor b of N, T_{Nb} 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.

To detect loops, the *Trust Manager* on N reuses the table of <the node id of a source node, forwarded sequence interval [a, b] with a significant length> (see Section 3.2) in the past few periods. 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_{oldNb} is the latest trust level value of b. We use a binary variable *Loop* to record the result of loop discovery: 1 if a loop is received; 0 otherwise. After the degradation, as in the update of energy cost, the new trust level of b is

$$T_{newNb} = (1 - w) \times T_{oldNb} + w \times Loop,\ w\ (0, 1),$$

Where w can be chosen by specific applications.

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 the past few periods with the broadcast messages from the base station about undelivered data. 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:

$$T_{newNb} = (1 - w) \times T_{oldNb} + w \times DeliveryRatio, w \in (0, 1),$$

Now, suppose an adversary M forges the identity of the base station by replaying all the routing packets from the base station. At first, it is able to deceive its neighbors into believing that M is a base station; as a result, M may attract a large amount of data packets, which never reach the base station. However, after the base station broadcasts the information about those undelivered packets, M's neighbors will downgrade M's trust level values in their neighborhood table. Note that M is only capable of replaying but is not capable of manipulating or generating authenticated broadcast messages, and that M usually cannot prevent other nodes from receiving a broadcast message from the base station. As time elapses, M's neighbors will start realizing that M is not trust-worthy and will look for other next-hop candidates that are more reliable. Similarly, if M forges the identity of another valid

appealing node, *M*'s neighbors will gradually realize that *M* is not reliable.

IV. IMPLEMENTATION AND EMPIRICAL EVALUATION

We have implemented a protocol based on A Trust Directing System for wireless sensor networks in TinyOS 1.x, which currently runs on mica2 motes. Both the authentication and encryption of packets reuse the implementation of TinySec [7] TinySec uses CBC mode encryption scheme with Skipjack as the block cipher and an authentication scheme based on a four-byte message authentication code (MAC) computed by the CBC-MAC construction procedure. The MAC field is computed over the whole message including all the headers; it also serves as the CRC field of the packet. Data encryption can be disabled. In a routing packet, the next-hop id is replaced by a neighborhood broadcast address or a network broadcast address to indicate that it is a neighborhood or whole network broadcast. The acknowledgement of data packets is enabled. Considering the fact that floating-point computation is not sup-ported by sensor hardware, the implementation uses an integer in [0, 100] to represent trust level; the update of energy cost and trust level values is also implemented using integer arithmetic's.

This implemented TARF protocol requires moderate program storage and memory usage. For comparison, we list the ROM size and RAM size requirement for this protocol and two other protocols on mica nodes in Table 1. The two other protocols are

Table 1.Size of protocol components implemented

Protocol	Authentication Encryption	ROM (bytes)	RAM (bytes)
Route	Tinysec	20696	1048
Min Route	Tinysec	22554	1990

named Route and MintRoute according to their directory name under TinyOS 1.x. Both Route and MintRoute were the "standard" routing protocols in TinyOS 1.x and make route decisions based on both link quality estimation and number of hops. Neither of these original protocols provides encryption or authentication; to compare on a fair basis, we also enabled the encryption and authentication mode of TinySec for Route and MintRoute. TinySec occupies 728 bytes of RAM and 7146 bytes of ROM [7]. Similarly to Route and MintRoute, this A Trust Directing System for wireless sensor networks protocol adopts energy-efficient routes in a misbehavior-free environment. However, with a comparable size, it also supports the circumvention of adversaries exploiting the replay of routing information, which is not provided by Route or MintRoute. Further, our experience shows that it is easy to incorporate this A Trust Directing System for wireless sensor networks protocol into most applications. As an example, we re-implemented the Surge application in the TinyOS 1.x directory with this A Trust Directing System for wireless sensor networks protocol. The program has a size comparable to that of the Surge implemented using Route or MintRoute.

To evaluate how effective A Trust Directing System for wireless sensor networks is against deception through replaying routing in-formation in the real world, we uploaded programs onto Motelab at Harvard University. As a public test bed of wireless sensor networks, at the time of our experiments, 184 TMote Sky sensor motes were deployed at 3 floors. These nodes are distributed among many rooms of the building, with an approximate indoor transmission of 100 meters. Approximately 14 nodes were removed, and nearly 50 nodes were disabled. Motelab switched its serial forwarder protocol from TinyOS 1.x to TinyOS 2.x and was equipped with TMote only Tmote Sky motes. Due to the unavailability of Tiny-Sec on TMote SKy nodes, we did not include authentication or encryption from Tiny-Sec in the uploaded programs. Further, considering the availability of routing protocols on TinyOS 2.x, we compared our TinyOS 2.x version of A Trust Directing System for wireless sensor networks with the collection tree routing protocol (CTP), which mainly employs link quality estimation in choosing next-hop nodes. Both protocols were integrated into a data collection application – Multihop Oscilloscope, which is named after its directory name in TinyOS 2.x. We configured the Multihop Oscilloscope to send out 5 samples in a single data packet every 5 seconds. The routing update occurred every 50 seconds. Because of the limited quota assigned by Motelab, our programs lasted maximally 30 minutes. Among all the nodes, one was chosen to be the base station. Another node was programmed to be a fake base station: it broadcast as if it were a base station but never delivered the received data to the real base station. The many experiments we executed indicate that our A Trust Directing System for wireless sensor networks protocol achieves at least 30% higher *throughput* than CPT when there is an "attractive" fake base station. Some fake base stations are not able to misdirect much traffic because they have a poor wireless connection with their neighbors and do not look "appealing".

Volume 5 Issue I, January 2017
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

In one experiment (Figure 4(a)), all nodes on the three floors were supposed to de-liver data to node 9 (the base station); node 15 (fake base station) replayed all the routing packets from the base station. By counting the data packets received at the real base station, A Trust Directing System for wireless sensor networks had approximately a 60% higher *throughput* than *CT P*. In another experiment (Figure 4(b)), only the nodes on the first floor (56 nodes totally) sent data to node 9 (the base station), and node 27 (fake base station) replayed the routing packets from the base station. As a result, A Trust Directing System for wireless sensor networks had approximately a 40% higher *throughput* than *CT P*.

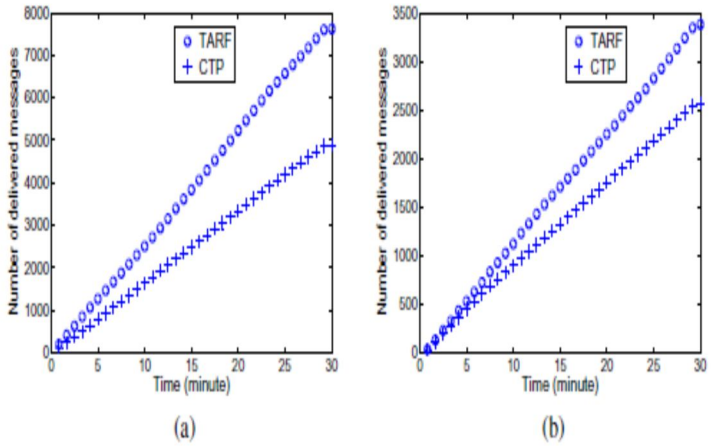

Fig 4. With a fake base at Motelab, (a) A Trust Directing System for wireless sensor networks had approximately a 60% higher *throughput* than *CTP* among 3 floors; (b) A Trust Directing System for wireless sensor networks had approximately a 40% higher *throughput* than *CTP* at a single floor.

We also recorded the number of redundant data packets received by the base station. It turns out that both A Trust Directing System for wireless sensor networks and CTP had redundancy ratios at no more than 2%. Though both CTP and A Trust Directing System for wireless sensor networks suppress redundant packets, a packet might be received more than once by the base station because an acknowledgment is lost when the route changes.

V. CONCLUSION

We propose A Trust Directing System for wireless Sensor networks, a trust-aware routing framework for WSNs, to secure multi-hop routing in WSNs against intruders exploiting the replay of routing information. With the idea of trust management, A Trust Directing System for wireless sensor networks enables a node to keep track of the trustworthiness of its neighbors and thus to select a reliable route. Not only does A Trust Directing System for wireless sensor networks circumvent those malicious nodes misusing other nodes' identities to misdirect network traffic, it also accomplishes efficient energy usage. Our implementation and simulation results indicate that (1) the efficiency of energy usage in A Trust Directing System for wireless sensor networks is generally at least comparable to that in existing protocols; (2) with the existence of traffic misdirection through "identity theft", A Trust Directing System for wireless sensor networks generally achieves a significantly higher *throughput* than other existing protocols; and (3) A Trust Directing System for wireless sensor networks is scalable and adaptable to typical medium-scale test bed environments and simulated conditions. Our future work is to further evaluate A Trust Directing System for wireless sensor networks with large-scale WSNs deployed in wild environments and to study how to choose parameters involved for specific applications. We believe that the idea of A Trust Directing System for wireless sensor networks can also be applied to general ad hoc networks and peer-to-peer networks to fight against similar attacks.

REFERENCES

[1] Al-Karaki, J., Kamal, A.Routing techniques in wireless sensor networks: a survey. IEEE Wireless Communications 11(6), 6–28 (2004)
[2] Blaze, M., Feigenbaum, J, Lacy J. Decentralized trust management. In: Proceedings of 1996 IEEE Symposium on Security and Privacy, pp. 164–173 (1996)
[3] Boukerche, A., El-Khatib, K., Xu, L., Korba, L.A novel solution for achieving anonymity in wireless ad hoc networks. In: Proceedings of the 1st ACM international workshop on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks, pp. 30–38 (2004)
[4] Ganeriwal, S., Balzano, L., Srivastava, M. Reputation based framework for high integrity sensor networks. ACM Trans. Sen. Netw. (2008)
[5] He, Q., Wu, D. Khosla, P. Sori: A secure and objective reputation-
[6] based incentive scheme for ad hoc networks. In: Proceedings of IEEE Wireless Communications and Networking Conference, pp. 825–830 (2004) .Kamvar, S., Schlosser, M., Garcia-Molina, H. The eigentrust algorithm for reputation management in p2p networks. In: Proceedings of the 12th international conference on

World Wide Web, pp. 640–651 (2003)
[7] Karlof, C., Sastry, N., Wagner, D. Tinysec: A link layer security architecture for wireless sensor networks. In: Proc. of ACM SenSys 2004 (November 2004)
[8] Karlof, C., Wagner, D. Secure routing in wireless sensor networks: attacks and counter measures. In: First IEEE International Workshop on Sensor Network Protocols and Applications (2003)
[9] Liang, Z., Shi, W. Pet: A personalized trust model with reputation and risk evaluation for p2p resource sharing. In: HICSS 2005: Proceedings of the Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS 2005) - Track 7. IEEE Computer Society, Los Alamitos (2005)
[10] Liu, A. Ning, P. Tinyecc: A configurable library for elliptic curve cryptography in wireless sensor networks. In: IPSN 2008: Proceedings of the 7th international conference on Information processing in sensor networks, pp. 245–256. IEEE Computer Society, Los Alamitos (2008)
[11] Liu, Z. Joy, A. Thompson, R.A dynamic trust model for mobile ad hoc networks. In: Proceedings of the 10th IEEE International Workshop on Future Trends of Distributed Computing Systems, pp. 80–85 (2004) .
[12] Mat lab, http://www.mathworks.com
[13] Motelab, http://motelab.eecs.harvard.edu
[14] Perrig, A. Szewczyk, R. Wen, W. Culler, D. Tygar, J.SPINS: Security protocols for sensor networks. Wireless Networks Journal (WINET) 8(5), 521–534 (2002)
[15] Wang, Y. Vassileva, J.T rust and reputation model in peer-to-peer networks. In: Proceedings of the 3rd International Conference on Peer-to-Peer Computing, p. 150 (2003)
[16] Watro, R. Kong, D. Cuti, S. Gardiner, C. Lynn, C. Kruus, P.Tinypk: securing sensor net-works with public key technology. In: SASN 2004: Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks, pp. 59–64. ACM, New York (2004)
[17] Woo, A. Tong, T. Culler, D. Taming the underlying challenges of reliable Multihop routing in sensor networks. In: Proceedings of the First ACM SenSys 2003 (November 2003)
[18] Wood, A. Stankovic, J. Denial of service in sensor networks. Computer 35(10), 54–62 (2002)
[19] Zhan, G. Shi, W. Deng, J.Poster abstract: Sensortrust - a resilient trust model for wsns. In: SenSys 2009: Proceedings of the 7th International Conference on Embedded Networked Sensor Systems (2009)