TRUST EVALUATION BASED SECURITY IN WIRELESS SENSOR NETWORK

Janani.C
M.E. Computer Science, III Year (Part Time)
SSM College of Engineering
Namakkal, India

P.Chitra
B.E., M.S., Ph.D.
Professor, SSM College of Engineering
Namakkal, India

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. The situation is further aggravated by mobile and harsh network conditions. 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, that has been designed and implemented TARF, a robust trust-aware routing framework for dynamic WSNs. Without tight time synchronization or known geographic information, This project provides trustworthy, time efficient 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 implementation and empirical experiments on large-scale WSNs under various scenarios including mobile and RF-shielding network conditions.

Keywords-: Wireless Sensor Networks, Wireless Sensor Network, Trusted Aware Routing Framework (TARF), TinyOS.

I. INTRODUCTION

Wireless sensor networks (WSNs) are ideal candidates for applications to report detected events of interest, such as military surveillance and forest fire monitoring. A WSN comprises battery-powered senor nodes with extremely limited processing capabilities. 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.

1.1. Problem Statement

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. 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, which is known as a wormhole attack.

A node in a WSN 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. It may drop packets received, forward packets to another node not supposed to be in the routing path, or form a transmission loop through which packets are passed among a few malicious nodes infinitely.

Sinkhole attacks can be launched after stealing a valid identity, in which 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." This same technique can be employed to conduct another strong form of attack Sybil attack: through replaying the routing information of multiple legitimate nodes, an attacker may present multiple identities to the network. A valid node, if compromised, can also launch all these attacks.

Time synchronization service in WSN has to meet challenges which are substantially different from those in infrastructure based networks. For instance, as each sensor has a finite battery source and communication is expensive in terms of energy, an important issue of WSN is *energy efficiency*. In addition, WSN show a higher failure probability over the time than in traditional networks due to battery depletion or destruction of the sensors, and changes in the environment can dramatically affect radio propagation causing frequent network topology changes and network partitions. Moreover, at high densities WSN become much more likely to suffer communication failures due to contention for their shared communication medium. These elements lead to strong *energy efficiency*, *self configuration* and *robustness* requirements. In the last few years several clock synchronization protocols for WSN have been proposed based on different approaches, such as the Reference Broadcast Synchronization (RBS) proposed by Elson et al., or hierarchical approaches, or interval{based, or probabilistic approaches for energy efficiency. However, despite their diversity, these proposals share a common viewpoint: they provide an accurate time estimate by means of *periodic* synchronization performed by each sensor node and based on messages exchanged with its neighbor nodes. Clearly, each clock adjustment is energy consuming since it involves transmitting messages and listening, besides the computational cost.

II Design Considerations

Before elaborating the detailed design of TARF, it would like to clarify a few design considerations first, including certain assumptions in Section *A* and the goals in Section *C*.

2.1. Assumptions

The target is 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, 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 acknowledgement packets, even remotely through a *wormhole*.
Additionally, to merely simplify the introduction of TARF to assume no data aggregation is involved.

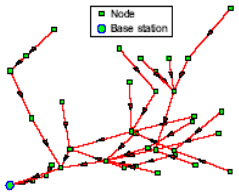


Fig. 1. Multi-hop routing for data collection of a WSN.

Nonetheless, this approach can still be applied to clusterbased 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 performing 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, 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 onehop transmission), the source id (the node that initiates the data), and the source's sequence number. It insists 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 onehop information to the base station. Also, it assumes the routing packet is sequenced.

2.2. 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 attackers, TARF may use TrustManager (Section 3.4) 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, μTESLA 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 are found in.

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 TinySec. It is possible that an adversary physically captures a non-base 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 TrustManager.

2.3. 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. This paper does not address the denial-of-service (DoS) attacks, where an attacker intends to damage the network by exhausting its resource. For instance, it does 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 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. Here the high *throughput* as one of our most important goals.

ENERGY EFFICIENCY Data transmission accounts for a major portion of the energy consumption. It evaluates 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 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 can measure the average hops that each delivery of a data packet takes, abbreviated as *hop-per-delivery*.

SCALABILITY & ADAPTABILITY TARF should work well with WSNs of large magnitude under highly dynamic contexts. It will evaluate the scalability and adaptability of TARF through experiments with large-scale WSNs and under mobile and hash network conditions. Here it does 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.

III MODULES DESCRIPTION

3.1. Routing the Network

In this module, the networks embedded on the physical fiber topology. However, assessing the performance reliability achieved independent logical links can share the same physical link, which can lead to correlated failures. Mainly, it focuses on assessing the reliability of energy level and trusted network.

3.2. Transfer File

In this module, Analysis the Shortest Path algorithm independently routes each logical link on a physical path with the minimum number of hops in trusted network basis. Hence, under the algorithm Shortest Path, each light- path greedily takes the most reliable route and transfers the file.

3.3. Sinkhole and Wormhole Attacks

Prevent the base station from obtaining complete and correct sensing data

Particularly severe for wireless sensor networks some secure or geographic based routing protocols resist to the sinkhole attacks in certain level many current routing protocols in sensor networks are susceptible to the sinkhole attack Set of sensor nodes

Continuously monitor their surroundings forward the sensing data to a sink node, or base station Many-to-one Communication vulnerable to the *sinkhole attack*, where an intruder attracts surrounding nodes with unfaithful routing information alters the data passing through it or performs selective forwarding

3.4. Energy Watcher & Trust Manager

In this module Cluster-based WSNs allows for the great savings of energy and bandwidth through aggregating data from children nodes and performing 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 subnetwork consisting of the cluster headers. Our framework can then be applied to this sub-network to achieve secure routing for cluster based WSNs.

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 [31], [32], [28]. In that case, when integrating TARF into those protocols with antiloop mechanisms, TrustManager may solely depend on the broadcast from the base station to decide the trust level can adopted such a policy when implementing TARF later. 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 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, it adopts the following mechanism to detect routing loops.

3.5. A Probabilistic Clock Reading Method

This method consists of two procedures:

- setVariables (Ti; H(ti)) invoked at each clock synchronization *ti*, with *Ti* accurate estimate of *ti* computed by running a clock synchronization protocol,
- *PCR*(*t*) that returns an estimate of the current reference time.

Figure 2 illustrates setV ariable(), and Figure 3 function
PCR(t). The data structures used are the followings:

- An array y[0..2] of records with two fields: y[j].val
  representing the observed value y_{i-j} relative to the time interval [t i-j , t i-j-1] for j = 0, 1, 2, and y[j].m = (t i-j –t i-j-1 / r) representing the times in which the sensor was unable to get its clock synchronized;
- An integer matrix M = M(3 x 4) of the coefficients of the linear system.

Fig 2:

$\mathbf{setVariables}(T_i, H(t_i))$:

1) $y[1] \leftarrow y[2]$
2) $y[2] \leftarrow y[3]$
3) $y[3].val \leftarrow T_i - H(t_i) - lastDev$
4) $lastDev \leftarrow T_i - H(t_i)$
5) $y[3].m \leftarrow round(\frac{H(t_i) - lastClock}{\Gamma})$
6) $lastClock \leftarrow H(t_i)$
7) $N \leftarrow N + 1$
8) if $N < \Psi$
9) updateMatrix()
10) else if $N = \Psi$
11) computeCoeff()

Fig 3:

PCR(t):

1) if $N < \Psi$
2) $\quad l \leftarrow y[1].m + y[2].m + y[3].m$
3) $\quad \beta[1] \leftarrow \frac{2}{3} - \frac{y[1].m}{l}$
4) $\quad \beta[2] \leftarrow \frac{2}{3} - \frac{y[2].m}{l}$
5) $\quad \beta[3] \leftarrow \frac{2}{3} - \frac{y[3].m}{l}$
6) $dev \leftarrow \beta[1]y[1].val + \beta[1]y[2].val + \beta[1]y[3].val$
7) $\Delta H \leftarrow H(t) - lastClock - dev$
8) $T \leftarrow T_i + \Delta H - dev\frac{\Delta H}{\Gamma}$
9) return T

3.6. 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, it would like to clarify a few important points on thedesign of EnergyWatcher and TrustManager. First, 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.

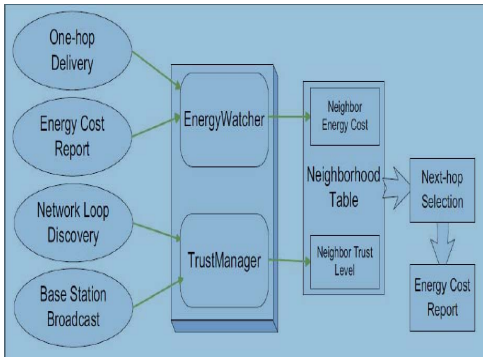


Fig. 4. Each node selects a next-hop node based on its neighborhood table, and broadcast its energy cost within its neighborhood. To maintain this neighborhood table, *Energy Watcher* and *TrustManager* on the node keep track of related events (on the left) to record the energy cost and the trust level values of its neighbors.

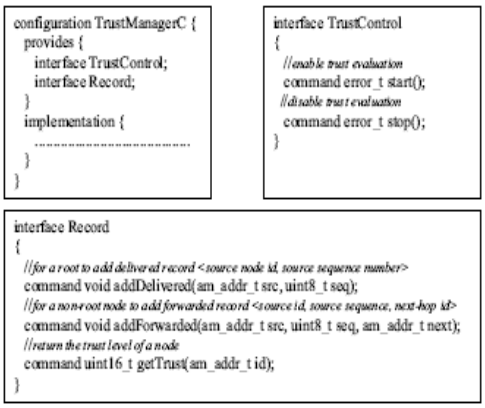
IV IMPLEMENTATION

In order to evaluate TARF in a real-world setting, it implemented the *TrustManager* component on TinyOS 2.x, which can be integrated into the existing routing protocols for WSNs with the least effort. Originally, it had implemented TARF as a self-contained routing protocol on TinyOS 1.x before this second implementation. However, decided to re-design the implementation considering the following factors. First, the first implementation only supports TinyOS 1.x, which was replaced by TinyOS 2.x; the porting procedure from TinyOS 1.x to TinyOS 2.x tends to frustrate the developers. Second, rather than developing a self-contained routing protocol, the second implementation only provides a *TrustMan ager* component that can be easily incorporated into the existing protocols for routing decisions. The detection of routing loops and the corresponding reaction are excluded from the implementation of *TrustManager* since many existing protocols, such as Collection Tree Protocol and the link connectivity-based protocol, already provide that feature. As it worked on the first implementation, it noted that the existing protocols provide many nice features, such as the analysis of link quality, the loop detection and the routing decision mainly considering the communication cost. Instead of providing those features, our implementation focuses on the trust evaluation based on the base broadcast of the data delivery, and such trust information can be easily reused by other protocols. Finally,

instead of using TinySec exclusively for encryption and authentication as in the first implementation on TinyOS 1.x, this re-implementation let the developers decide which encryption or authentication techniques to employ; the encryption and authentication techniques of TARF may be different than that of the existing protocol.

4.1. TrustManager Implementation

The *TrustManager* component in TARF is wrapped into an independent TinyOS configuration named TrustManagerC. TrustManagerC uses a dedicated logic channel for communication and runs as a periodic service with a configurable period, thus not interfering with the application code. Though it is possible to implement TARF with a period always synchronized with the routing protocol's period that would cause much intrusion into the source code of the routing protocol. The current TrustManagerC uses a period of 30 seconds; for specific applications, by modifying a certain header file, the period length may be re-configured to reflect the sensing frequency, the energy efficiency and trustworthiness requirement. TrustManagerC provides two interfaces (see Figure 4), TrustControl and Record, which are implemented in other modules. The TrustControl interface provides the commands to enable and disable the trust evaluation, while the Record interface provides the commands for a root, i.e., a base station, to add delivered message record, for a non-root node to add forwarded message record, and for a node to retrieve the trust level of any neighboring node. The implementation on a root node differs from that on a non-root node: a root node stores the information of messages received (delivered) during the current period into a record table and broadcast delivery failure record; a non-root node stores the information of forwarded messages during the current period also in a record table and compute the trust of its neighbors based on that and the broadcast information. Noting that much implementation overhead for a root can always be transferred to a more powerful device connected to the root, it is reasonable to assume that the root would have great capability of processing and storage.



A root broadcasts two types of delivery failure record: at most three packets of significant undelivered intervals for individual origins and at most two packets of the id's of the origins without any record in the current period. For each origin, at most three significant undelivered intervals are broadcast. For a non-root node, considering the processing and memory usage overhead, the record table keeps the forwarded message intervals for up to 20 source nodes, with up to 5 non-overlapped intervals for each individual origin. Our later experiments verify that such size limit of the table on a non-root node produces a resilient TARF with moderate overhead. The record table on a node keeps adding entries for new origins until it is full. With our current implementation, a valid trust value is an integer between 0 and 100, and any node is assigned an initial trust value of 50. The weigh parameters are: wupgrade = 0.1, wdegrade = 0.3. The trust table of a non-root node node keeps the trust level for up to 10 neighbors. Considering that an attacker may present multiple fake id's, the implementation evicts entries with a trust level close to the initial trust of any node. Such eviction policy is to ensure that the trust table remembers those neighbors with high trust and low trust; any other neighbor not in this table is deemed to have the initial trust value of 50.

V INCORPORATION OF TARF INTO EXISTING PROTOCOLS

To demonstrate how this TARF implementation can be integrated into the exiting protocols with the least effort, it 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.

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. 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, 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.

```
//Step 1. traverse the neighborhood table for an optimal candidate for the next hop
optimal_candidate = NULL
//the cost of routing via the optimal candidate provided by the existing protocol, initially infinity
optimal_cost = MAX_COST
//the trust level of the optimal candidate, initially 0
optimal_trust = MIN_TRUST
for each candidate in the neighborhood table
    if link is congested, or may cause a loop, or does not pass quality threshold
        continue
    better = false
    if candidate.trust >= optimal_trust && candidate.cost < optimal_cost
        better = true
    //prefer trustworthy candidates
    if candidate.trust >= TRUST_THRESHOLD && optimal_trust < TRUST_THRESHOLD
        better = true
    if candidate.trust >= ESSENTIAL_DIFFERENCE_THRESHOLD + optimal_trust
        better = true
    //effective when all nodes have low trust due to network change or poor connectivity
    if candidate.trust >= 3 * optimal_trust / 2
        better = true
    //add restriction of trust level requirement
    if candidate.trust >= TRUST_THRESHOLD && candidate.trust / candidate.cost >
optimal_trust / optimal_cost
        better = true
    if better == true
        optimal_candidate = candidate
        optimal_cost = candidate.cost
        optimal_trust = candidate.trust

//Step 2. decide whether to switch from the current next-hop node to the optimal candidate found:
if optimal_trust >= currentNextHop.trust                                    \
 || currentNextHop.trust <= TRUST_THRESHOLD                                 \
 || current link is congested and switching is not likely to cause loops    \
 || optimal_cost + NEXTHOP_SWITCH_THRESHOLD < currentNextHop.cost           \
    currentNextHop = optimal_candidate
```

Fig. 5. Routing decision incorporating trust management.

This new implementation integrating TARF requires moderate program storage and memory usage. Here implemented a typical TinyOS data collection application, MultihopOscilloscope, based on this new protocol. The MultihopOscilloscope application, with certain modified sensing parameters for our later evaluation purpose, periodically makes sensing samples and sends out the sensed data to a root via multiple routing hops. Originally, MultihopOscilloscope uses CTP as its routing protocol. Now list the ROM size and RAM size requirement of both implementation of MultihopOscilloscope on non-root Telosb motes in Table 1. The enabling of TARF in MultihopOscilloscope increases the size of ROM by around 1.3KB and the size of memory by around 1.2KB.

VI Related Work

It is generally hard to protect WSNs from *wormhole* attacks, *sinkhole* attacks and *Sybil* attacks based on identity deception. The countermeasures often requires either tight time synchronization or known geographic information. FBSR, as a feedback-based secure routing protocol for WSNs , uses a statistics-based detection on a base station to discover potentially compromised nodes. But the claim that FBSR is resilient against *wormhole* and *Sybil* attacks is never evaluated or examined; the Keyed-OWHC-based authentication used by FBSR also causes considerable overhead. There also exists other work on trust-aware secure routing that is evaluated only through computer simulation, such as.

There are certain existing secure routing solutions for WSNs based on trust and reputation management; however, they rarely address the "identity theft" exploiting the replay of routing information. Two such representative solutions are ATSR and TARP. Neither ATSR nor TARP offers protection against the identity deception through replaying routing information. ATSR is a location-based trust-aware routing solution for largeWSNs. ATSR incorporates a distributed trust model utilizing direct and indirect trust, geographical information as well as authentication to protect the WSNs from packet misforwarding, packet manipulation and acknowledgements spoofing. Another trust-aware routing protocol for WSNs is TARP, which exploits nodes' past routing behavior and link quality to determine efficient paths.

VII Conclusions And Future Work

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. This prospective has a noticeable impact on WSN for their strong energy efficiency, robustness and self configuration requirements.

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 static and mobile settings, hostile network conditions, as well as strong attacks such as wormhole attacks and Sybil attacks.

References

[1] Guoxing Zhan, Weisong Shi, "Design and Implementation of TARF: Trust-Aware Routing Framework for WSNs" IEEE Transactions On Dependable And Secure Computing, vol. 9, no. 2, March/April 2012

[2] 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.

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

[4] A. Wood and J. Stankovic, "Denial of Service in Sensor Networks," Computer, vol. 35, no. 10, pp. 54-62, Oct. 2002.

[5] 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.

[6] 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.

[7] I. Krontiris, T. Giannetsos, and T. Dimitriou, "Launching a Sinkhole Attack in Wireless Sensor Networks; The Intruder Side," Proc. IEEE Int'l Conf. Wireless and Mobile Computing, Networking and Comm. (WIMOB '08), pp. 526-531, 2008.

[8] J. Newsome, E. Shi, D. Song, and A. Perrig, "The Sybil Attack in Sensor Networks: Analysis and Defenses," Proc. Third Int'l Conf. Information Processing in Sensor Networks (IPSN '04), Apr. 2004.

[9] L. Zhang, Q. Wang, and X. Shu, "A Mobile-Agent-Based Middleware for Wireless Sensor Networks Data Fusion," Proc. Instrumentation and Measurement Technology Conf. (I2MTC '09), pp. 378-383, 2009.

[10] S. Ganeriwal, L. Balzano, and M. Srivastava, "Reputation-based framework for high integrity sensor networks," ACM Trans. Sen.Netw., 2008.

[11] G. Zhan, W. Shi, and J. Deng, "Design, implementation and evaluation of tarf: A trust-aware routing framework for dynamic wsns," http://mine.cs.wayne.edu/_guoxing/tarf.pdf, Wayne State University, Tech. Rep. MIST-TR-2010-003, Oct. 2010.

[12] Daniela Tulone, "A Resource–efficient Time Estimation for Wireless Sensor Networks," Department of Computer Science University of Pisa

.