

Location-centric Isolation of Misbehavior and Trust Routing in Energy-constrained Sensor Networks

Sapon Tanachaiwiwat¹, Pinalkumar Dave¹, Rohan Bhindwale², Ahmed Helmy¹

1. Department of Electrical Engineering 2. Department of Computer Science

University of Southern California, Los Angeles, CA. 90089

{tanachai, pdave, bhindwal, helmy} @usc.edu

Abstract In sensor networks a large number of distributed sensors collaborate to deliver information to the sinks. Such scenario assumes trust between sensor nodes. However, sensors may fail or be compromised (in military operations) in a way that renders them misbehaving. In this work we target a misbehavior model in which a misbehaving node participates in routing signaling while consistently dropping queries and data packets. We target static sensor networks in which geographic routing is used. We identify and study the route infection effect in which one misbehaving node may block the path to many nodes in a sensor network.

We propose a novel, location-centric, architecture for isolating misbehavior and establishing trust routing in sensor networks. Our scheme fits the data-centric nature of sensor networks and is suitable for use in energy-constrained networks. Much of our protocols operate in the sinks relieving the sensors from a lot of functionality. Our protocols select trusted paths that do not include misbehaving nodes, by identifying insecure locations and routing around them efficiently via detour points using embedded blacklists and modified geographic or trajectory routing. For insecure location discovery we propose efficient one-shot probing. Cheat-proofing is achieved using location correlation to remove false reporting. Our simulations show how our scheme effectively increases the throughput and energy-efficiency of a sensor network and alleviates the effect of route infection in geographic routing.

Keywords Secure Location, Sensor Networks, Trust Routing

1. INTRODUCTION

In many wireless sensor networks, a large number of distributed sensors must collaborate to deliver the requested information to the sink(s). Such scenario assumes trust relationships between sensor nodes. However, in critical and sensitive mission such as military operations, sensors may fail or be compromised in a way that renders them malicious or (at least) non-cooperative. In this work we target a misbehavior model in which a compromised or faulty sensor node consistently drops data packets while participating in signaling and routing protocols. Our work targets static sensor networks in which geographic routing is used.

Existence of misbehaving nodes in location-aware networks may be quite harmful, since the misbehaving node may be chosen on the routing path from the sink(s) to many sensor nodes. We call such paths *infected routes*. The percentage of

geographic routes infected by the existence of misbehaving nodes may exceed, by far, the percentage of misbehaving nodes in the network. Figure 1 further illustrates and quantifies such *route infection* effect. As shown, existence of only 5% of misbehaving nodes leads to infection of more than 60% of the routes in a grid sensor network (more than 35% for randomly placed sensors). With 10% misbehaving nodes, 88% of the grid routes are infected (54% for random topology) so on.

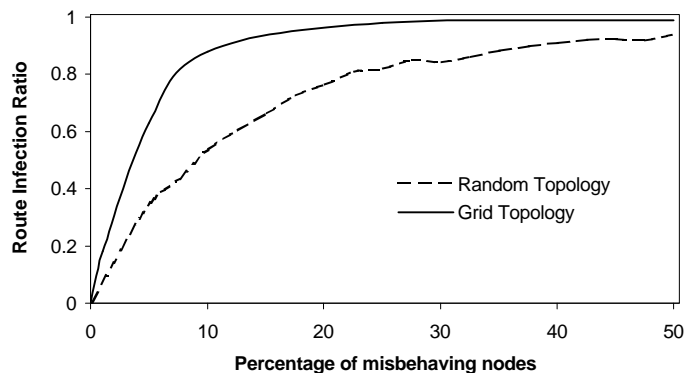


Figure 1. Route infection effect in geographic routing due to existence of misbehaving nodes. (Simulations shown for 3700 node grid topology with ave. 8 neighbors per node, and random topology with 1000 nodes and ave. 20 neighbors per node, with one sink in the middle of the topology)

This route infection effect motivates our work on a new location-centric architecture for identification and isolation of misbehaving nodes using trust routing and avoidance of insecure locations. Our architecture is location-centric (as opposed to node-centric), which fits the data-centric model [6] for sensor networks, and the stateless nature of geographic routing. The two main components of our architecture are: (i) trust routing and (ii) insecure location discovery and isolation. We propose the novel concept of insecure locations that are identified, probed and isolated by the sink(s). We also introduce a trust routing protocol for our architecture, where we aim to select trusted paths that do not include misbehaving nodes by identifying the insecure locations and routing around them efficiently using modified geographic or trajectory routing. Black lists of insecure locations are embedded in the packet header instead of being broadcast. This approach scales well by keeping record of locations instead of sensor nodes.

Our trust routing mechanism uses asymmetric authentication scheme μ TESLA [11] to form the chain of trust. Our trust

routing protocol also investigates the behavior of the nodes en route by using a compact trust routing table that records the cooperativeness of locations. For insecure location discovery we propose and compare several probing techniques; expanding TTL search and one-shot. We show how using these techniques, along with location information also helps in cheat-proofing the system, against false location reporting from misbehaving nodes.

Several aspects of our architecture address the energy-constrained nature of sensor networks.

(1) One energy-efficiency metric is the bits delivered per unit energy consumed in the network. Hence, increasing the throughput in the network increases the energy-efficiency. By alleviating the route infection effect we show that the throughput is increased dramatically.

(2) Our one-shot probing mechanism used for misbehaving location identification incurs less number of transmissions than related schemes and hence is more energy-efficient.

(3) Instead of adopting the usual approach of broadcasting the blacklist, we use blacklist embedding in the packet header to guide the packet through detour points to route around the misbehavior.

Our design considers trust stability, in which we demonstrate that events of congestion or normal packet loss do not lead to blacklisting of nodes. This is achieved by careful choice of the *trust encouragement* parameter. Our results show that our protocols are effective in reducing the effect of route infection by increasing the network throughput. Also, our scheme incurs no extra overhead when the network is secure (i.e., in absence of misbehaving nodes).

The rest of this paper is organized as follows. Section 2 discussed related work. Section 3 provides an architectural overview. Section 4 presents the trust routing protocol (*TRANS*). Section 5 introduces the mechanisms for identifying and isolating insecure locations. Section 6 provides evaluation results, and Section 7 concludes.

2. RELATED WORK

Security issues in sensor networks have been addressed in several areas [6][11] such as Denial-of-Service attacks [14]. Some have proposed security mechanisms for sensor networks [10][17] as well as intrusion tolerance protocol [2]. Work in [10] [11] provides mechanisms for secure communication suitable for sensor networks: μ TESLA for authenticated broadcast and SNEP protocols for one-to-one confidentiality. The security protocols for sensor networks (SPINS) architecture does not address the problem of compromised sensors. Several effects of malicious behavior were identified in [6] including bogus routing information, selective forwarding, sinkholes, Sybil, wormholes, HELLO floods, and Ack spoofing. In this work, we identify an effect called route infection in which existence of one misbehaving node infects (i.e., invalidates) many routes in geographic routing. We propose a new location-centric approach to secure location-aware sensor networks.

Work on secure ad-hoc networks includes Ariadne [3], SEAD [4] that introduce the concept of attack behavior model for ad hoc networks and apply the Tesla (earlier version of μ TESLA) technique for broadcast authentication to reduce the overhead for asymmetric cryptographic calculation based on loose time synchronization between each host. We adopt this model for low overhead hop-by-hop authentication between sensor nodes. In [16] the secure AODV protocol is proposed to improve the security of AODV routing protocol in ad hoc networks. SAR [15] introduces the security attributes as parameters into ad hoc route mechanism. By using SAR, the node can discover the route with greater security guarantee. The route discovered may not be the shortest path route but it is guaranteed to be a secure route. In [14] schemes are proposed to solve problems of key management in ad hoc networks such as threshold cryptography that deals with sets of compromised nodes, and web of trust where each host has a certificate repository and shares some part of its repository to form the certificate chain. Also mechanisms are identified for finding trust in probabilistic model where the trust value decreases if the negotiation fails and increases on success. Some of our trust table mechanisms resemble that work. We address issues of selecting parameters of increase/decrease of trust and attempt to relieve the sensor nodes by moving much of the functionality to the sink(s).

All the above schemes are node-centric in which each node needs to perform similar tasks to maintain security. In data-centric sensor networks, however, where sensor nodes are resource and energy constrained, this approach may not be suitable. Our approach, by contrast, uses a location-centric approach and proposes much of the security tracking functionality at the sink(s) with some distributed help from the sensors for hop-by-hop authentication. We use simple modifications to existing geographic and trajectory routing techniques to route around the insecure locations.

Other related work lies in the area of misbehavior identification and isolation. Following is a brief discussion of related methods on misbehavior identification.

Binary Search [1][14] mechanisms probe nodes along a suspected path using inputs from intermediate nodes and an expanding ring probing. This phase discovers faulty links on the path from the source to the destination in $O(\log n)$ probes, where n is the average length of the path. A black list of the malicious hosts is broadcast via trusted neighbors until it reaches the neighbor of that malicious host. In our study we evaluate and compare binary search, expanding exponential TTL and *one-shot* schemes for probing. Our simulation shows that on average our *one-shot* scheme is the most efficient. Instead of expansive broadcasting of blacklists we propose to either embed the blacklist (of insecure locations) in the packet header and perform modified geographic routing, or geocast the blacklist to the neighborhood of the insecure location.

Watchdog [10] detects misbehaving nodes by overhearing transmission. It maintains a buffer of recently sent packets and compares each overheard packet with the packet in the buffer to see if there is a match. If a packet remained longer

than timeout, then it increases a failure tally for the responsible node or if the tally exceeds a threshold, the node is determined to be misbehaving and the source will be notified. Its advantage is that it can detect misbehavior at the forwarding level. However it might fail in the presence of ambiguous collisions. It cannot identify the false report from misbehaving node. Most importantly, sensor networks may employ sleep/wakeup duty cycle to conserve energy, in which case overhearing may not be possible and may lead to excessive energy wastage. Our approach does not rely on overhearing. Moreover we propose a scheme to detect false reporting of locations by correlating various reports from neighboring locations.

Pathrater [10] is designed for use in avoiding routing packets through misbehaving nodes (reported by Watchdog). Each node maintains a rating for every other node it knows within the network. It calculates a path metric by averaging the node ratings in the path. The metric gives a comparison of the overall reliability of different paths. If there are multiple paths to the same destination, it chooses the path with the highest metric. Our approach by contrast uses stateless geographic routing that is more suitable and efficient for sensor networks than stateful, path-based, routing. Also, aggregation of insecure-locations allows our scheme to scale well as the number of compromised nodes increase, as opposed to having to keep track of misbehaving nodes.

3. ARCHITECTURAL OVERVIEW

The goal of our architecture is to ensure safety of information requested from sink in the presence of misbehaving nodes. By safety, we mean that the queries and replies should be delivered with certain reliability (i.e., minimum delivery rate) if alternative, secure, paths are available. Typically, this means that sink and source packets (i.e., queries and replies) must be monitored by the sink and intermediate relay nodes. For energy efficiency, monitoring is done only as necessary; i.e., by nodes participating in the forwarding path.

3.1. Architectural Components

TRANS is a security mechanism built on top of geographic routing composed of (i) Trust Routing and (ii) Insecure Location Avoidance. We shall discuss how trust routing is efficiently achieved for a large collection of sensors in Section 4. In Section 5, we shall discuss how to avoid the insecure locations (which might be occupied by multiple misbehaving or compromised nodes) using trust values and blacklisting. While we focus on communication to a single sink, the architecture is easily applied for multiple sinks. Figure 2 depicts the architecture corresponding to our approach, where TRANS mechanisms are implemented using distributed TRANS modules consisting of (1) *Trust Routing Module* (TRM) installed in the sink and all the sensor nodes, and (2) *Insecure Location Avoidance Module* (ILAM) only installed in the sink node.

Cryptography verification, packet forwarding observation, availability observation modules in TRM are designed to monitor the behavior of sensors with which the sink

communicates. The trust table processor interprets these behaviors to trust values for their peer. Black list processor determines whether it is the detour point in finding alternate route. ILAM functions are required in the sink(s) only.

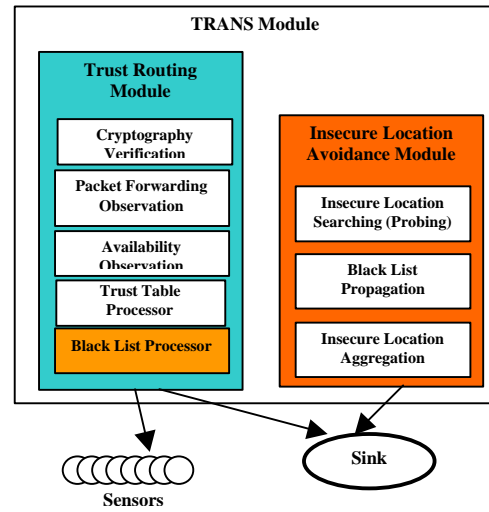


Figure 2 The TRANS Architecture building blocks

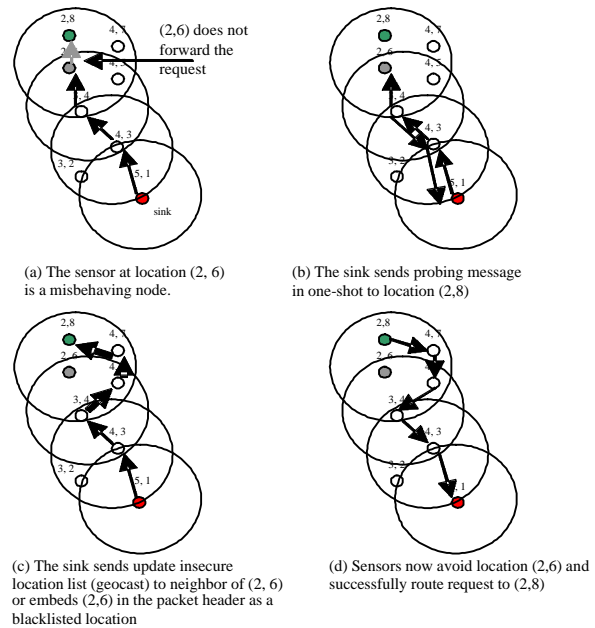


Figure 3 Example scenario for identification and isolation malicious node at (2,6)

To illustrate the synergy among the above architectural components, we shall walk through an example scenario. As part of the normal operation of the sensor network the sink sends queries systematically to a specific location using geographic routing (or to aggregate locations using geocast), packets are sent over secure and available sensors (initially established through cryptography verification and availability observation). When the sink does not receive replies to its queries it invokes the packet forwarding observation module that decides whether probing should be performed (after exceeding a certain delivery loss threshold). If probing is

necessary it is performed using a low overhead search mechanism to identify misbehaving nodes. Once misbehavior location has been identified (after removing false location reports) it is merged into the list of insecure locations (blacklist) at the sink and may be potentially aggregated into a group of insecure locations. Then the blacklist is embedded into future packets to route around insecure locations using detour points or geographic trajectory routing. Figure 3 provides a scenario for identification, isolation and routing around a misbehaving nodes (or location) in a location-aware sensor network using our proposed architecture.

3.2 Assumptions

For architecture we make the following assumptions:

- Availability of (approximate) location information and the ability to perform geographic (or trajectory) routing.
- The sink is always trusted and cannot be compromised. Although we provide our study examples based on a single sink network, our architecture applies to the multiple sink case.
- Dense distribution of sensors and availability of alternative geographic routes.
- Misbehavior model of consistent dropping of queries and data packets, while potentially participating in routing signaling.
- Query replies from sensors carry location information.
- All traffic flow is from/to the sink(s).

We believe all of these assumptions are reasonable for location-aware sensor networks.

4. TRUST ROUTING FOR LOCATION-AWARE SENSOR NETWORKS (TRANS)

We propose a new routing protocol, *TRANS*, to account for the non-cooperation and malicious behavior of sensors efficiently. *TRANS* uses the concept of trust to select a secure path and avoid insecure locations [13]. We assume sensors know their (approximate) locations and that geographic routing (e.g., GPSR [3]) is used. We also assume that all destination nodes use loose-time synchronization asymmetric mechanism μ TESLA to authenticate all requests and that the shared encryption key will be carried with the Message Authentication Code (MAC) from sink or base station to ensure request integrity and confidentiality. Based on initial authentication, each sensor initializes trust values for its neighbors' locations. A trusted neighbor is a sensor that can decrypt the request and has enough trust value (based on forwarding history as recorded by the sink and other intermediate nodes). A sink sends a message only to its trusted neighbors (i.e. its trust value is higher than specified trust threshold) for the destined location. Those neighbors correspondingly forward the packet to their trusted neighbors that have the nearest location to destination. Thus the packet reaches the destination along a path of trusted sensors.

In our architecture, blacklisting is distributed (or embedded in data packet) by the sink and we assume that the sink will

not be compromised. The sink identifies misbehavior (by observing replies), probes potential misbehaving locations, and isolates insecure locations. To prevent blacklist tampering, we can use the MAC as we use it for request.

The route selected by *TRANS* may not be optimal in terms of hop count but provides safe and unaltered delivery of data. The sensors and sink monitor the activities of their neighbors and adjust their trust values accordingly. After excessive packet drops, the sink initiates a search for insecure locations along the path. On discovery of such location the sink records the insecure location. This information is later used to either route the packet through detour points inserted in the packet header, or to propagate a location blacklist (via geocast) that helps route around the insecure locations using standard geographic routing.

Specifying and dynamically adjusting the location trust parameters present an interesting research challenge in our design. If these parameters are not set carefully, cooperative location may be tagged as misbehaving due to their proximity to insecure location. Especially in the presence of lossy wireless channels and possible congestion events, a mechanism may misconstrue normal packet loss for misbehavior. We introduce the trust *encouragement* factor ***b*** to prevent blacklisting due to abrupt network dynamics and to dampen oscillations in trust value adjustments.

4.1 Basic Secure Location Concept

We define a secure location as a location free of misbehaving sensor nodes and is safe for packets traversing from sink to source and vice versa. A secure location excludes any insecure location that holds any misbehaving nodes.

Define: T = Total Area of Sensor Coverage

S = Secure Locations (Areas)

S' = Aggregated Insecure Locations (Areas)

Hence $S = T - S'$...(1)

In general, the larger S' the greater the effect of route infection and the less the probability of delivering a packet using regular geographic routing.

Aggregated Insecure Location Format: There may be many ways to represent the list of insecure locations for inclusion in the black list. Here, we specify the insecure location that we want to avoid (or the black list of locations) as follows:

- **Single location:** [(x1, -), (y1, -)]
- **Group location:** [(x1, x2), (y1, y2)]

Insecure locations can be aggregated to form a *group location* to save the storage and bandwidth. A *group location* may be simply represented in rectangular form for simple encoding. Insecure locations can be known by a probing mechanism and trust value of neighbor sensors. The single location representation grows linearly as the list of insecure locations grows. Group location representation, on the other hand, scales well as the aggregation probability increases with increase in the list of insecure locations.

4.2 Trust Table and Related Security Mechanisms

Based on the replies a sink receives from various locations in the network, locations are assigned trust values. Also sensor node calculates trust values for its neighbors' location. The trust values are set and adjusted based on trust parameters, measurements populating a trust table, the malicious node isolation protocol, and, optionally, sharing blacklists (using chain-of-trust concept). The sink node assigns the shared key for the entire sensor network. For energy saving purposes only the destination authenticates the sink request and only the sink authenticates the data from the destination. Each sensor authenticates every neighbor sensor, however it only monitors *availability* and *packet forwarding* of the sensors in the forwarding path.

The trust table contains the following entries: Cryptography (C), Availability (A), Packet forwarding (P) and Trust value (T). For sensor i , the Trust value can be calculated from the product of C_i , A_i and P_i . If the trust value is below a specific trust threshold, then this location is considered insecure and is avoided when forwarding packets (e.g., by using the second nearest location, using detour points in modified GPSR or using an alternative route-around approach such as Trajectory Based Forwarding [4]).

Following we elaborate on the trust table entries.

- **Cryptography (C):** Each sensor in the network is assumed to have the same cryptographic capability (e.g., MAC function). Sensors supporting cryptography for encryption are given a higher trust value ($C=1$), and are able to authenticate the sink's messages unless compromised.

- **Availability (A):** This value is based on beacon measurements and is used to account for node or link failures or wakeup/sleep schedules in sensor networks. Alternatively, a Hello mechanism for liveness checks between sensors obviates the need for availability measurements. Tracking wakeup/sleep cycle can also identify the abnormality of each sensor (we assume that each sensor know each other wakeup/sleep schedule). This can be used alternatively for authentication purpose.

The dynamic *availability* of location i , defined as A_i , is calculated as follows based on a window of ' n ' beacons.

$$A_i = \frac{\sum_{j=1}^n QA_j}{n} \quad \dots(2)$$

where QA_j represents j^{th} beacon; if the sensor is not available then $QA_j = 0$, otherwise $QA_j = 1$.

Note that non-availability is never used to blacklist a location as insecure, but is simply used to avoid using that location while the sensor(s) are un-available.

- **Packet forwarding (P):** When a request reaches a sensor at or near the destination location, the sensor replies to the requester (or sink) acknowledging the packet reception and reporting the location. When the source (or sink) receives the reply it increases the trust values for the locations en route to the destination.

The dynamic *packet forwarding* value for location i , defined as P_i , is calculated as the running average of a window of m replies, as follows.

$$P_i = \frac{\sum_{j=1}^m QP_j}{m} \quad \dots(3)$$

where QP_j represents j^{th} reply status; if the request/reply is received then $QP_j = 1$, otherwise $QP_j = 0$.

- **Trust value (T):** The trust value is the value that represents the overall trustworthiness of each location. Only locations with trust value higher than specified *trust threshold* will be selected to participate in data forwarding. The trust value of location i , defined as T_i , is a function of C_i , A_i , P_i and b (described next) for that location as follows.

$$T_i = C_i \cdot A_i \cdot b \cdot P_i \quad \dots (4)$$

Table 1 Summary of Parameter Characteristics

	Value	Type	Equation	Measure
Cryptography(C)	0,1	Int	No = 0, Yes =1	Authentication
Availability (A)	0-1	Ratio	Beacon replies/Beacon sent	Periodic Beacon
Packet Forwarding (P)	0-1	Ratio	Successful packet Fwd/ Total packet Fwd	Reply of the destination
Trust Value (T)	(0-1)	Ratio	($C \cdot A \cdot P$)	($C \cdot A \cdot P$)

Our main packet monitoring approach borrows from watchdog, but, we cannot assume that neighbors are always awake and overhearing the packets (This is considered power-wasteful and it is not suitable for sensor network).

4.3. Encouraging Factor (b)

The multiplication factor of P is called encouragement factor (b). The encouragement factor helps to encourage the packet forwarding in the initial phase of the packet forwarding, where P may oscillate with packet loss for first few samples. The value of b is initially set high and decreases with increase in number of samples. The values for b were chosen to dampen oscillations in trust value and absorb effects of loss of initial packets. We have set the values of b empirically, based on simulations, as follows.

Table 3 Trust values based on encouragement factor

Count	C	A	P	b	$T = C \cdot A \cdot (P \cdot b)$
1	1	1	0.1	2.0	0.2
2	1	1	0.2	1.8	0.36
3	1	1	0.3	1.6	0.48
4	1	1	0.4	1.4	0.56
5	1	1	0.5	1.2	0.6
6	1	1	0.6	1.0	0.6
7	1	1	0.7	1.0	0.7
8	1	1	0.8	1.0	0.8
9	1	1	0.9	1.0	0.9
10	1	1	1	1.0	1.0

As per our different experiment when the values of n and m are equal to (or more than) 10, the trust values are set

appropriately. We do not observe any abrupt rise or fall in the trust values.

4.4 Routing Mechanism

In many data-centric sensor networks, only the sink initiates data transfer using queries. The sink creates a message with its location, the destination location, authentication message (MAC) and (optionally) detour locations. It encrypts this message with its shared key and sends it using geographic routing. Only those neighbors that the sink trusts and who know its shared key will be able to decrypt the request. The appropriate trusted neighbor decrypts the request, adds its location, encrypts the message with its share key and sends it to trusted neighbors that are near the destination. This way the packet traverses a path of trusted neighbors using location-aware routing. After the sink's request is authenticated, the destination creates a reply with its location information encrypted with the shared key and sends it back to the sink using geographic routing (avoiding insecure locations embedded in the packet) as shown in Figure 3.

5. ISOLATING INSECURE LOCATION

Consider a case when the sink measurements indicate that replies from a certain location are consistently dropped. Subsequently, the packet forwarding value in the trust table for that location will drop steadily. If the trust value (based on forwarding packets) drops below a certain threshold this indicates a potential misbehaving insecure location, so the sink procedures to isolate such location using *probing*. We propose and study several schemes for probing to identify and isolate insecure locations, including (a) expanding TTL ring search, (b) binary search and (c) one shot.

```

Start: i = 1, n = 0, m = 0
latest_hop = 1;
previous_hop = 0;
fwd_loop:
send probe to latest_hop
if acked then
{ latest_hop = latest_hop + 2^n
n = n++ until latest_hop == destination
previous_hop = latest_hop
badnode_fwd = latest_hop
goto fwd_loop }
else
{ if( badnode_fwd == badnode_bck)
{ malicious_node = badnode_fwd
exit }
m = 0
goto bck_loop
bck_loop:
send probe to latest_hop - 2^m
m = m++ until latest_hop == previous_hop
badnode_bck = latest_hop
if not acked then
{ if( badnode_fwd == badnode_bck)
{ malicious_node = badnode_bck
exit }
goto bck_loop }
else
{ n = 0
goto fwd_loop
}
End:

```

Figure 4 Pseudo code for E-TTL at the sink

In expanding TTL search (E-TTL) the sink sends probe packets with increasing hop-count (See Figure 4). Each intermediate node decrements the hop-count before forwarding. When the hop count reaches zero at a node, that node sends Ack to the sink informing it of its location and that the packet was received safely. Hence, the sink identifies that part of the path as safe and increases the hop count in subsequent packets. Alternatively the TTL can also be increased exponentially rather than linearly, which incurs less delay than basic E-TTL, and may also be restricted to a small number. In binary search, as the name implies, the probe is sent along the path in a binary search fashion and on average incurs $O(\log n)$ requests and replies where n is the number of nodes along the path. The last scheme we present is *one-shot*, where the sink node sends only one probe message along the path to which each node en route replies with its location. This scheme reduces the number of packets sent and reduces discovery delays. Figure 5 illustrates one-shot probes.

```

Start:
Send probe to the closest neighbor
Ask sensors en-route receiving the probe to
1. reply to the sink
2. forward to next closest neighbor
While ( t < time_out)
{ Wait for Ack
Collect the ack from sensor
Add to the list
}
Find the ack from farthest hop from the list
malicious_node = farthest hop + 1
End:

```

Figure 5 Pseudo code for One-Shot at the sink

Figure 6 compares the number of request/reply transmissions in the network for binary search, E-TTL and one-shot schemes. This directly reflects the energy spent in identifying and isolating misbehavior. This network has 8 hops and we simulated malicious node at first hop till seventh hop. In almost all cases, one-shot perform better than the others. E-TTL seems least suitable when malicious node is far from the sink. In Section 5, we use the one-shot scheme.

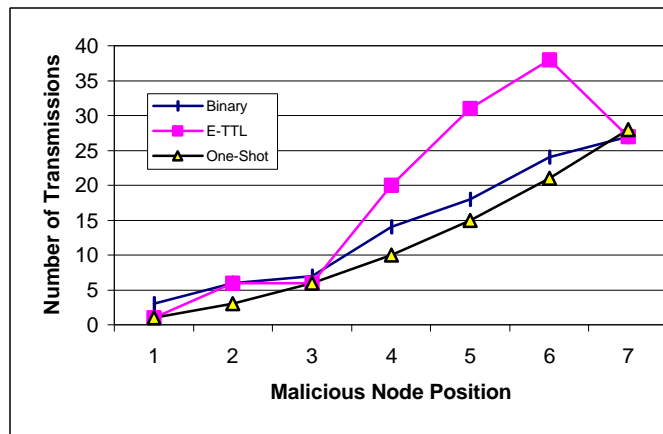


Figure 6 Number of Operation Comparison between Binary Search, E-TTL and One-Shot

Note that a misbehaving node may or may not respond to the probe but according to our model it will not forward it. This may be interpreted to mean that the misbehaving node is either: (1) the farthest node along the path to respond or (2)

the first node on the path not to respond. To increase confidence level in identification multiple probes may be used from different directions (detour points). Details omitted for brevity. Also note that the sink is the only one to initiate the search for misbehaving nodes and the identification of insecure location. As mentioned earlier that this may be easily applies to the multiple sink case.

We introduce two schemes for isolating and blacklisting insecure locations: (a) black list geocast or (b) embedded black list (using detour points). In the first scheme the sink geocasts (e.g., using LAR or [18]) the black list to the neighborhood of the insecure location. This approach does not require the modification of geographic routing or the packet header because the non-cooperative node (at the insecure-location) will be simply removed from the neighbor list and will not be selected to participate in any routing activity. In the second approach the sink includes the black list information in the header of the packet and sends it directly to a detour point using geographic routing. This approach incurs less packet overhead but requires modification of packet headers and possible simple extensions to geographic routing to route via detour points.

6. EVALUATION AND RESULTS

We consider the following parameters for evaluation: packet delivery ratio, added overhead, and goodput. Overhead is measured in queries, requests/replies transmissions for isolation of misbehavior and black list propagation (as a worst-case performance for our scheme we used geocast to the whole network, to obtain a conservative estimate). Goodput is defined as the ratio of number of acked packets to number of packets sent by source (including overhead packets). The simulations show that our architecture using the *TRANS* with one-shot/black-list flooding over grid topology with single insecure location can improve the goodput of network by 40% when compared to the same scenario without trust routing. It also shows that our protocol does not incur extra overhead if there is no misbehavior.

We use our discrete event simulator. We ran the simulator in two modes; when *TRANS* was “on” and when *TRANS* was “off”, for multiple topologies (ring, grid and random).

Ring Topology

In the ring topology one node (source) in the network sends data to its diametrically opposite node (sink). Along the path there is a misbehaving node. When *TRANS* is *off* all the packets get dropped and the data transfer never completes. In case of *TRANS on*, the protocol notices that packets are dropped and uses another path to reach the destination. Clearly the overhead in *TRANS on* is increased. This is shown by Figure 7. The goodput is much higher for *TRANS on*. Overall energy-efficiency (as bits/Joule) is increased.

We ran tests for various number of nodes and observed that the overhead increases as the size of network increases. This is understandable as more nodes become involved in the malicious node identification and thus the load on the network increases. Similarly the goodput decreases as there are more overhead packets in the network.

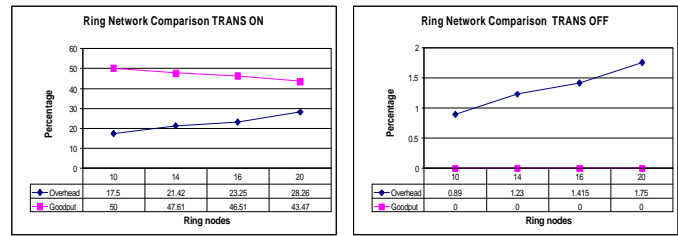


Figure 7 Performance Comparisons between TRANS ON and TRANS OFF for Ring Topology

Grid Topology

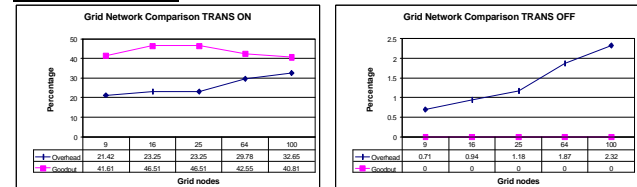


Figure 8 Performance Comparisons between TRANS ON and TRANS OFF for Grid Topology

In the grid topology the traffic pattern was chosen between nodes in the node in the left top corner and the node (sink) in the right bottom corner. One node along the diagonal of the network is misbehaving. Both *TRANS ON* and *TRANS OFF* select the diagonal as the best path. *TRANS ON* continues trying to send data through the infected path even though the malicious node along the path drops the data packets. *TRANS ON* is more intelligent and selects an alternate route and the goodput is increased drastically.

We again created varying number of nodes in the network and measured the metrics. We found that as before the overhead increases as the size of network increases (Fig 8). Similarly the goodput decreases as there are more overhead packets in the network. And again we attribute both these to the increase in the number of overhead packets that travel in the network.

Random Topology

We created random topologies and initiated multiple random data transfers (with multiple senders and receivers). Again we ran for *TRANS ON* and *TRANS OFF*. Greedy geographic routing (i.e., GPSR [7] without perimeter mode) was used.

We created 5 random topologies of 30 nodes and initiated 10 data transfers in them. Some of the data transfers happened to pass through infected routes. We observed that *TRANS ON* was able to complete all data transfers. *TRANS OFF* managed to complete data transfers on paths that did not contain malicious nodes. This is shown in Figure 9.

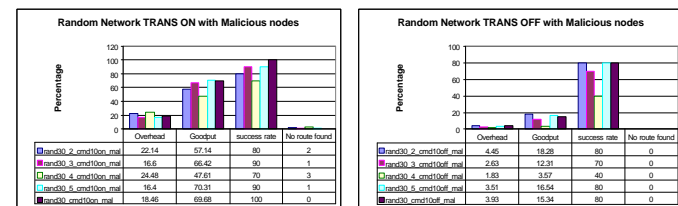


Figure 9 Performance Comparisons between TRANS ON and TRANS OFF for Random Topology with malicious nodes.

Comparing *TRANS ON* and *OFF* we observe that *TRANS ON* is always at least as good as *TRANS OFF* when their

goodputs are compared. Also the TRANS ON goodput is a healthy 80+ % in most cases. Comparing the overhead we again note that TRANS ON overhead (~20%) is higher than TRANS OFF (~3%). This is mainly due to the global geocast in our conservative (worst case) evaluation. Using embedded blacklist in the packet header or limited geocast is expected to reduce the overhead dramatically to be comparable to the TRANS OFF case.

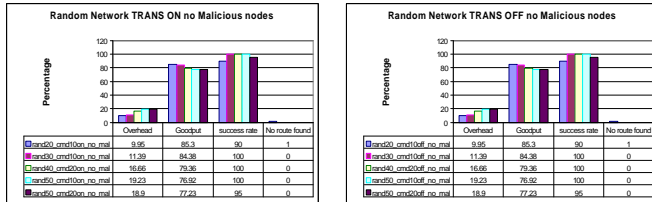


Figure 10 Performance Comparisons between TRANS ON and TRANS OFF for Random Topology with no malicious nodes

Interestingly we also ran some simulations where we created random topologies and started multiple data transfers with no malicious nodes in the network. Obviously now all the data transfers except the no routes should succeed. We then did a comparison of TRANS ON and TRANS OFF's overhead and goodput. We observed that when there are no malicious nodes in the network TRANS ON adds no overhead to the regular routing protocol. It completely follows TRANS OFF in all the metrics we observed.

7. CONCLUSION

We studied the effect of misbehavior in location-aware sensor networks and illustrated the drastic effects of route infection. This motivated our location-centric architecture for trust routing in sensor networks. The two main components of our architecture include the trust routing protocol (*TRANS*) and the misbehavior isolation and black listing. *TRANS* uses a location trust table with parameter adjustment mechanisms that dampen effects of packet loss and link failure. Functionality requiring active probing and black listing runs on the sink(s) only. A lightweight one-shot probing mechanism is used to identify the location(s) of misbehavior. Two energy-efficient schemes are proposed for black listing using embedded black listing or geocast.

Our simulations show that our architecture achieves node isolation, increased robustness against transient failures, and alleviates route infection effects. With the use of our architecture results indicate significant improvement in goodput (and hence energy-efficiency metrics) in the presence of misbehaving nodes, while no extra overhead is incurred over regular geographic routing with no misbehaving nodes.

In sum, the main contribution of our work include: (a) the illustration of the route infection problem in geographic routing, (b) the explicit investigation of the design trade-off between secure/trusted routing and greedy geographic routing, and (c) the introduction of efficient node isolation and black listing schemes.

REFERENCES

- [1] B. Awerbuch ,D. Holmer, C. Nita-Rotaru ,H. Rubens, "An on-demand secure routing protocol resilient to byzantine failures", International Conference on Mobile Computing and Networking, Proceedings of the ACM workshop on Wireless security 2002 , Atlanta, GA, USA
- [2] J. Deng, R.Han,and S. Mishra, "Security Support for In-Network Processing in Wireless Sensor Networks", ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '03) October 2003.
- [3] Y. Hu, A. Perrig, and D. Johnson, "Ariadne: A secure on-demand routing protocol for ad hoc networks," in The 8th ACM International Conference on Mobile Computing and Networking, MobiCom 2002.
- [4] Y. Hu, D. Johnson, A. Perrig, "SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks", IEEE Wkshp on Mobile Computing Systems & A plications (WMCSA), June 2002
- [5] C. Intanagonwivat, R. Govindan and D. Estrin "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks" ACM MobiCOM, August 2000.
- [6] Chris Karlof and David Wagner, "Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures", IEEE International Workshop on Sensor Network Protocols and Applications, May 2003.
- [7] B. Karp, H. T. Kung "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks" ACM MobiCOM, Aug 2000.
- [8] D. Niculescu, B. Nath, "Trajectory-based forwarding and its applications" ACM MobiCOM 2003.
- [9] S. Marti, T.J. Giuli, K. Lai, M. Baker "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks", ACM MobiCOM, Aug 00.
- [10] A. Perrig, R. Szewczyk, V. Wen, D. Culler, J. D. Tygar, "SPINS: Security Protocols for Sensor Networks", ACM MobiCOM, July 2001.
- [11] S. Slijepcevic, M. Potkonjak,V. Tsitsis, S. Zimbeck, M. B. Srivastava.On communication Security in Wireless Ad-Hoc Sensor Network Eleventh IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'02) June 10 - 12, 2002 Pittsburgh, Pennsylvania, USA.
- [12] Verma, R., O'Mahony, D. & Tewari, H., NTM- Progressive Trust Negotiation in Ad Hoc Networks, in Proceedings of the First Joint IEI/IEE Symposium on Telecommunications Systems Research, Dublin, November 27th, 2001, 8pp
- [13] W. Wang, Y. Lu, B. K. Bhargava , On Security Study of Two Distance Vector Routing Protocols for Mobile Ad Hoc Networks, *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, March 2003
- [14] Anthony D. Wood, John A. Stankovic. Denial of Service in Sensor Networks. *IEEE Computer*, 35(10):54-62, 2002
- [15] S. Yi, P. Naldurg, and R. Kravets , Security-Aware Ad Hoc Routing for Wireless Networks, , Poster presentation, ACM Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc), Long Beach, California, October, 2001.
- [16] M. G.Zapata, Secure ad hoc on-demand distance vector routing, *ACM SIGMOBILE Mobile Computing and Communications Review* Volume 6 , Issue 3 (July 2002)
- [17] S. Zhu, S. Setia and S. Jajodia. LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks. *ACM Conference on Computer and Communications Security (CCS '03)*, October, 2003.
- [18] K. Seada, A. Helmy, Efficient Geocasting with Perfect Delivery in Wireless Networks", *IEEE Wireless Communications and Networking Conference (WCNC)*, March 2004.