The Interleaved Authentication for Filtering False Reports in Multipath Routing based Sensor Networks

Youtao Zhang [1,3], Jun Yang [2], Hai T Vu [3]

[1] Computer Science Department
University of Pittsburgh
Pittsburgh, PA 15260
zhangyt@cs.pitt.edu

[2] Computer Science and Engineering Department
University of California at Riverside
Riverside, CA 92507
junyang@cs.ucr.edu

[3]Computer Science Department
University of Texas at Dallas
Richardson, TX 75083
{zhangyt,htv041000}@utdallas.edu

Abstract

In this paper, we consider filtering false reports in braided multipath routing sensor networks. While multipath routing provides better resilience to various faults in sensor networks, it has two problems regarding the authentication design. One is that, due to the large number of partially overlapped routing paths between the source and sink nodes, the authentication overhead could be very high if these paths are authenticated individually; the other is that false reports may escape the authentication check through the newly identified node association attack. In this paper we propose enhancements to solve both problems such that secure and efficient authentication can be achieved in multipath routing. The proposed scheme is (t+1)-resilient, i.e. it is secure with up to t compromised nodes. The upper bound number of hops that a false report may be forwarded in the network is $O(t^2)$.

1 Introduction

The wireless sensor network has emerged as a promising computing model for many applications including those running in hostile environments e.g. tracking enemy targets in a battlefield. Since sensor nodes are usually left unattended after deployment, they are vulnerable to varying forms of security attacks [4]. Once a sensor node is captured, the sensitive information stored in the node is exposed. In addition, the compromised node may be used to launch further attacks.

Several recent schemes have been proposed to defend the false report injection attack in the sensor network [10, 13, 9]. In such an attack, a compromised node injects false reports or modifies its relayed reports. Without detecting such reports, the sink may reach a suboptimal or even wrong decision. In addition, routing false reports to the sink consumes the limited energy of relay nodes on the routing path and reduces the lifetime of the network. In the schemes proposed in [10, 13, 9], the en-route authentication strategy is employed such that false reports are detected and dropped early in the routing path to save the routing energy and prolong the lifetime of the network. The authentication is performed between two nodes that share the same authentication key. The key can be set up by randomly selecting from a key pool [10], or creating pairwise authentication keys in an interleaved approach [13], or combining location information [9], or periodically refreshing the keys [12]. While each relay node has limited authentication ability, a number of consecutive sensor nodes can confidently detect and drop false reports in several hops.

In this paper, we consider filtering false reports in a multipath routing based sensor network. Multipath routing has gained its popularity as it adapts better to various faults such as sensor failure and signal conflicts. In multipath based routing, the source node sends the reports back to the sink along several paths that may and may not be disjoint. Even some relay nodes on some paths failed, the report can still be routed to the sink as long as there exists one well-behaved path. Studies have showed that several braided (partially disjoint) paths can achieve better tradeoff among factors such as packet delivery, energy consumption and failure node recovery [1].

In a sensor network with braided multipath routing [1], the number of possible paths between the sink and source

1-4244-0054-6/06/$20.00

nodes are typically large. The false report detection is hence complicated as dropping them along one path does not effectively stops the attack. False reports may escape the authentication check either through a different path or through malicious node association. While the interleaved authentication for single path routing [13] can be adopted, the overhead is probably high. To address this problem, we make the following contributions in this paper:

- We identify a new type of attack – the node association attack in either the single path routing or the multipath routing scenario. This attack is more severe in a multipath routing environment due to dynamic node re-association. We elaborate and explain the attack using examples. We then propose to defend the attack with sufficient information included in the ACK message at the node association stage.
- We propose a $(t+1)$-resilient interleaved authentication method for multipath routing. Nodes are associated conservatively in multipath routing. We maintain similar en-route authentication overhead and show that the false report can be forwarded at most $O(t^2)$ hops where t is the number of nodes in the shortest path.

The remainder of the paper is organized as follows. Section 2 reviews the related work and in particular the interleaved hop-by-hop authentication [13] and the braided multipath routing [1] that our scheme are based on. Section 3 details the problems that we target to solve. Section 4 presents our algorithm and its security analyses. Section 5 concludes the paper.

2 Background

2.1 False reports filtering

In this section we first briefly describe the state-of-art false report filtering schemes and then describe in more detail the interleaved hop-by-hop authentication (IHA) [13] for single path routing.

Ye *et al.* [10] proposed a statistical en-route filtering scheme in which each node randomly picks up a subset of keys from a global key pool. A report is endorsed with multiple such keys and authenticated by relay nodes who have at least one of endorsing keys. The false report is dropped if a mismatch is found. Zhu *et al.* [13] proposed IHA to associate nodes on a routing path interleavingly such that each node can authenticate the MAC generated by its association node. Yang *et al.* proposed a scheme [9] to achieve high resilience in terms of the number of compromised nodes through incorporating location information in generating authentication keys. By periodically updating the keys with the help from neighboring nodes, Zhang *et al.* [12] proposed a scheme to minimize the harm of compromised nodes.

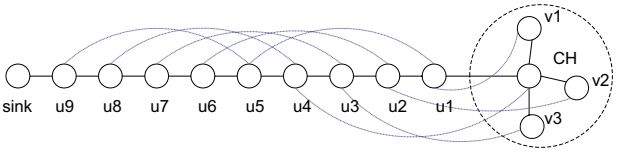


Figure 1. The IHA scheme (graph adapted from [13], resilient to up to 3 compromised nodes (t=3)).

We now briefly describe the IHA scheme [13] in which nodes are associated and MACs are checked within association pairs. Figure 3 depicts such an association that can achieve $(t+1)$ resilience where $t = 3$. Nodes $v1$ to $v3$ are nodes within a cluster with the cluster head CH. $u1 - u9$ are relay nodes that forward reports from the CH to the sink. Basically, a node is associated with an upstream (towards the sink) and a downstream (towards the source) node of $t+1$ hops away. For example, $u5$ is associated with $u9$ and $u1$. A unique pairwise key is used in each association, e.g., the $u1 - u5$ pair uses $K_{u1.u5}$ and $u5 - u9$ pair uses $K_{u5.u9}$. When $u5$ receives a report, it authenticates the MAC generated by $u1$ using $K_{u1.u5}$. Upon success, $u5$ replaces this MAC with a new one using $K_u 5.u9$. The new MAC is to be authenticated by $u9$. As we can see, a report needs to carry sliding $t+1$ MACs computed from keys corresponding to $t+1$ association. If any t nodes in this path are compromised, i.e., t keys are exposed, the last association will guarantee that a faulty report be detected because its key is still hidden. To be complete, the nodes with less than $t+1$ hops from the cluster head are associated with distinct nodes from within the cluster, e.g., $u1$ is associated with $v1$. The nodes apart from the sink for less than $t+1$ hops do not need to associate with upstream nodes since there are less than $t+1$ of them and they are unable to conspire to fool the sink. A false report in IHA can travel $O(t^2)$ hops in the network.

2.2 Multipath routing

The design of different routing techniques in wireless sensor networks (WSNs) is largely influenced by non-traditional factors [5] such as energy consumption, network dynamics, data reporting/aggregation model, fault tolerance etc. For example, directed diffusion routing [3], geographic adaptive fidelity routing [8] forward packets to a subset of nodes towards the sink such that the energy consumption is reduced and the network lifetime is prolonged.

Multipath routing schemes [1, 6, 11] maintain multiple alternative paths between the sink and the source nodes in order to adapt to link congestion and node failures in the network. Next, we briefly review the *braided multipath routing* which achieves better failure resilience and energy consumption compared to disjoint multipath routing schemes.

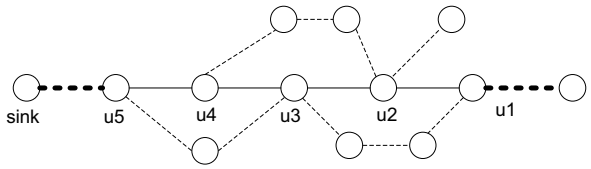


Figure 2. The braided multipath routing.

Developed from the directed diffusion routing scheme [3], braided multipath routing [1] maintains a primary path and several alternative paths between the sink and the source (cluster head) nodes. In Fig. 2, the primary path is shown by the solid line and the alternative paths are shown by dashed lines. When the sink is to reinforce the routing paths from the sink to the cluster head, each node on the primary path identifies an alternative next node in addition to the one on the primary path; each node on the alternative path only identifies one next node, in the same way as the directed diffusion routing. The alternative paths may join the primary path partially such that the primary and alternative paths are braided with nodes overlapping with each other. As we can see, the number of possible paths between two distant nodes is large. In Fig. 2, for example, there are five possible paths from node $u1$ to node $u5$. Ganesan *et al.* [1] showed that the number is proportional to the n^{th} Fibonacci number if there are n nodes on the primary path. The braided multipath scheme achieves better resilience to node failures simply because the high number of possible paths from the source to the sink. We will explain next our design based on this scheme.

3 Problem Statement

In this section we elaborate the two problems that we are to solve in this paper. Before the discussion we define some terms that we use in the paper.

Definition 1 *Two nodes V_1 and V_2 are* directly *associated if they are associated with each other for authentication.*

Definition 2 *Two node V_1 and V_2 are* indirectly *associated if there is a non-empty list of nodes [X_1,...,X_n] such that V_1 is directly associated with X_1, X_i is directly associated with $X_{i+1}(1 \leq i \leq (n-1))$, and X_n is directly associated with V_2.*

Definition 3 *A node V_1's* authentication chain*, abbreviated as V_1-chain, consists of all nodes that are directly and indirectly associated with V_1.*

Definition 4 *If there are r number of associated between two associated nodes V_1 and V_2, we say V_1 is $(r+1)$* rounds *away from V_2.*

For example in Fig. 1, nodes $v3$, $u3$ and $u7$ are on the $v3$ authentication chain. nodes $u3$ and $u7$ are directly associated while nodes $u7$ and $v3$ are indirectly associated. Node $u7$ is two rounds away from $v3$.

3.1 Problem 1: malicious node association manipulation

The effectiveness of the interleaved hop-by-hop (IHA) authentication scheme relies on the correctness of the node association, i.e., the (t+1) MACs can be generated and verified alternatively. In the original scheme the node association is established at the beginning stage of each epoch of transferring sensor reports. Fig. 3 depicts the association process. A HELLO message is first sent from the sink to the cluster head CH (source node), followed by a reply ACK message from CH to the sink. Each message contains a *node list* consisting of up to $(t+1)$ nodes that are used in the node association. These nodes are the $(t+1)$ nearest neighbors in one direction of any node in the path. Hence, the list is a sliding window of the nodes that the HELLO/ACK message passes through.

When the HELLO message is propagated from $u8$ to CH in Fig. 3, a node list grows from an empty set in $u8$ to a set of i, $i \leq (t+1)$ nodes that indicates the last i nodes visited in that order. A node N finds its up-stream associated node M by reading the head of the list. The list is then modified by removing M and inserting N at the end, and passed on to the next node. Note that nodes within $(t+1)$ hops from the sink has no up-stream association nodes. When CH receives the HELLO message, it replies back with an ACK message by forming a node list containing itself and t sensors in its cluster. Those sensors are to be associated with relay nodes that are within $t+1$ hops from CH. A node N finds its down-stream associated node S by reading from the tail of the list. The list is then updated by removing S and inserting N at the beginning, and passed on to the next node. The nodes in pair associated in this way are $(t+1)$ hops away from each other.

Unfortunately the important node association phase is not rigorously protected from malicious attacks. A compromised node on the route can manipulate the node list and deceive the nodes down in the path. This is serious since, due to the varying nature of sensor routing, nodes in sensor networks may have to be re-associated after each epoch. At this point some sensor nodes may have already been compromised and thus can initiate the attack to perform malicious node association manipulation.

Next we elaborate the attacking procedure using an example in Fig. 4 where t=4. That is, if there are four compromised nodes $X1$, $X2$, $X3$, and CH, the scheme should still be secure. The HELLO and ACK messages also include a node list with 5 node IDs. For clarity, we omit the HELLO messages received by $u3-u8$ and the ACK messages beyond $u6$ as they are not relevant in the attack. Ideally the two nodes in each association pair should be 5 (=t+1) hops away from each other.

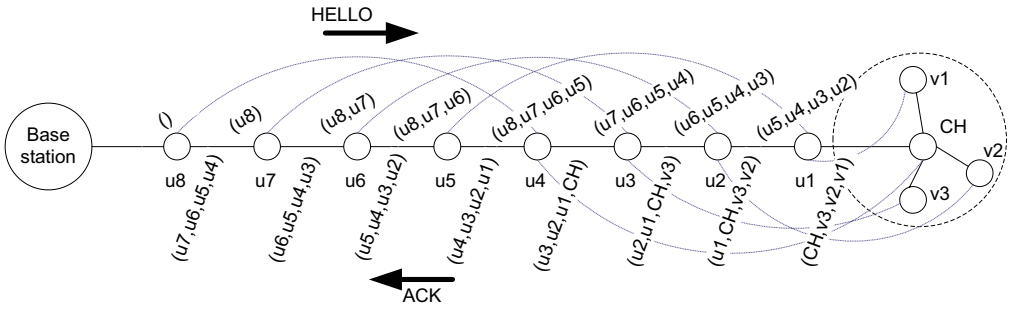


Figure 3. The node association in IHA scheme.

Let us focus on $X3$ who manipulates the node association. $X3$ first receives a HELLO message with the list $(u7, u6, u5, u4, u3)$. Therefore $X3$ is up-stream associated with $u7$. However, since it is compromised, it sends $u2$ a fake node list $(u5, u4, u3, u6, X3)$. This list ensures that $u2$ and $u1$ are up-stream associated with $u5$ and $u4$ respectively while the correct association should be $u6$ and $u5$. Similarly, in processing the ACK message, $X3$ forges the outgoing node list to $u3$. The fake node list ensures that $u3$ to $u7$ are down-stream associated with CH, $u1$, $u2$, $X2$, and $X3$ respectively. In addition, we see that $X1$ and $X2$ are down-stream associated with $u2$ and $u1$ respectively, due to the initial malicious node list formed by CH.

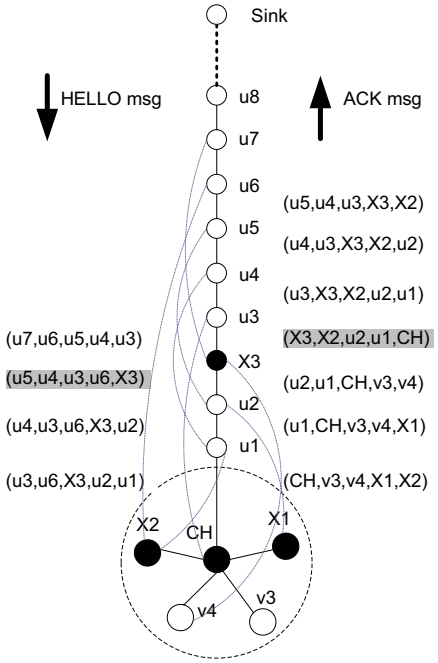


Figure 4. The node association attack (t=4).

After the above malicious association, the compromised CH can send any false reports to the sink without being detected en-route. The false report can be constructed through the collaboration between $X3$ and CH. In a false report, suppose P is the report content; $XMACP$ is the XOR-MAC to be authenticated by the sink [13]; the rest are five pairwise keyed MACs for en-route authentication. Let $key_{X2.u1}$ denote the pairwise key between node $X2$ and $u1$. The corresponding MAC – $\text{MAC}_{key.X2.u1}(\text{P})$ is generated by $X2$ and to be verified by $u1$. Clearly, if one node in an association pair is compromised, the MAC can always be forged. Let $MACx$ be some arbitrary unimportant bits in a MAC.

The report forged by CH is

$$[\,P, XAMCP, \quad MAC_{key.X2.u1}(P), MAC_{key.X1.u2}(P), MACx, MACx, MACx]$$

This false report can pass $u1$ and $u2$ since the corresponding MACs are generated by compromised nodes $X1$ and $X2$. After passing these two nodes, two new MACs are added – $MAC_{key.u1.u4}(P)$ and $MAC_{key.u2.u5}(P)$ which are to be validated by $u4$ and $u5$ respectively.

Therefore the compromised node $X3$ can rearrange the report and generate the new report as follows.

$$[\,P, XMACP, \quad MAC_{key.CH.u3}(P), MAC_{key.u1.u4}(P), MAC_{key.u2.u5}(P), MAC_{key.X2.u6}(P), MAC_{key.X3.u7}(P)]$$

In this report, the 1st, 4th and 5th MAC can be generated since each corresponding association has a compromised node. The 2nd and 3rd MACs are received from $u1$ and $u2$. At this point, all five MACs are consistent with the false report content. It can be forwarded to the sink without being detected.

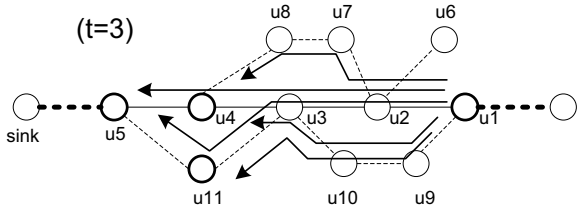
Initial observation. By observing this attack, it is not difficult to find that it is the node $X2$ that creates the problem because it sits on two authentication chains – the $u1$-chain and $u2$-chain. This effectively reduces the number of different MACs to t, or it is possible to construct (t+1) different MACs from t compromised nodes. We will devise a mechanism to defend such attacks in section 4.

3.2 Problem 2: security and overhead tradeoffs in multipath routing

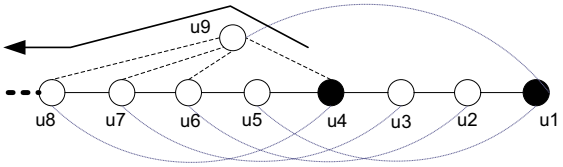
As false reports may also be injected in a multipath routing based sensor network, it is equally important to perform en-route authentication and drop false reports as early as possible.

However with braided multipath routing, there are large number of different routing paths between the sink and the source nodes. In [1] Ganesan *et al.* showed that the number is proportional to the n^{th} Fibonacci number if there are n nodes on the primary routing path. The maintenance overhead would be prohibitively high if the IHA scheme is directly applied to each possible path.

Instead a reasonable solution is to consider multiple routing paths simultaneously. To perform the node association at each node, only its neighboring nodes need to be considered. The IHA associates two nodes that are (t+1) hops away from each other into one pair, and uses a node list containing the last traveled (t+1) nodes to help node association. In multipath routing, we can similarly consider a short routing path segment. The difference is that there might be multiple (t+1)-hop-away nodes in each routing direction. It is not trivial to design a secure and efficient node association scheme accordingly.



(a) Multiple (t+1)-hop-away nodes



(b) The 3rd attempt: assign each into one association group

Figure 5. The node association in multipath routing (t=3).

Let us discuss three attempts with different overhead and security levels. The first attempt is that a node sets up a different pairwise key with each of its (t+1)-hop-away nodes; authentication is then performed based on which path the report is routed along. For example Fig. 5(a) illustrates this attempt with t=3. There are five subpaths from $u1$ to $u5$,

$path_1$: $u1-u2-u3-u4-u5$
$path_2$: $u1-u2-u7-u8-u4-u5$
$path_3$: $u1-u9-u10-u3-u4-u5$
$path_4$: $u1-u9-u10-u3-u11-u5$
$path_5$: $u1-u2-u3-u11-u5$

Thus $u1$ creates three up-stream pairwise keys — one with $u5$ for $path_1$ and $path_5$, one with $u4$ for $path_2$ and $path_3$; and one with $u11$ for $path_4$. In addition, $u1$ has to create similar number of down-stream pairwise keys (not shown in the figure). This attempt is secure as each different subpath is independently protected by IHA. However, as one can see, the overhead is still very high. In addition to the storage for saving and distinguishing multiple keys, a relay node (e.g. $u1$) has to generate and transmit three different MACs up-stream, instead of one MAC per node in IHA. The latter is more problematic as data transmission in sensor networks consumes more energy. The authentication overhead may increase m times in multiple routing if each node generates m MACs on average.

To reduce the authentication overhead, we can group all up-stream (t+1)-hop-away nodes together using one authentication key. Similarly, a different authentication key is created for all down-stream (t+1)-hop-away nodes. In the above example, a group authentication key is created among $u1$, $u5$, $u4$, and $u11$. Thus no matter which path the packet is to take, $u1$ only needs to generate one up-stream MAC. Unfortunately while this scheme can greatly reduce the overhead, the security is compromised. For example, due to the existence of $path_2$, node $u5$ also sets up a key with $u2$ as they are (t+1) hops away from each other. Therefore, $u5$ stays on two different authentication chains ($u1$-chain and $u2$-chain). If $u5$ is compromised, it can generate two MACs such that (t+1) legal MACs can be constructed by $u5$ and other (t-1) compromised nodes. In summary, if a node gets involved in two different authentication chains, it is possible to break the authentication protection by generating (t+1) legal MACs from t compromised nodes.

Based on the above observation, the third mechanism could be to allow each node to stay on only one authentication chain. In the above example, if $u5$ has been associated with $u1$, it may not be associated with $u2$ even with the existence of the path $path_2$. However, the difficulty is that between $u1$ and $u2$, which node should be associated with $u5$. We may run into security problems if we give the priority to one over the other. For example, if the primary path is of higher priority in Fig. 5(b), then $u1$ to $u4$ should be associated with $u5$ to $u8$ respectively. However, no matter how we associate node $u9$, only two different MACs are checked along the path $u4-u9-u8$. In such a setting even if a false report is detected and dropped along the path $u4-u5-u6-u7$, it might still reach $u8$ via the shortcut $u4-u9-u8$. If this continues to happen beyond $u8$, a false report can reach the sink without being dropped en-route.

Therefore, it is challenging to design an interleaved authentication scheme that is both secure and efficient for multipath routing.

3.3 Assumptions

Before presenting our algorithm, we discuss the network and attack models that we consider in the paper.

The network model We consider a sensor network that consists of a number of battery-powered sensor nodes and

a sink node with abundant resources, e.g. energy and computation power. We assume the sink node cannot be compromised. Each sensor is assigned with a unique ID and a secret key before deployment. Both the ID and the key are known to the sink node. Sensor nodes are left unattended after deployment. They monitor events of interests and send the data reports back to the sink. When an event happens in the network, it can be detected by multiple nodes in a cluster. We assume that majority of sensing nodes for any single event are trustworthy. We assume the clustering technique is used since it has been proven effective in reducing energy consumption of the entire sensor network [2, 7]. Data reports are first sent to the cluster head who will construct an aggregated report that also contains the IDs and MACs from sensing sensors. We use the multi-hop braided multipath routing scheme [1]. Thus the report is forwarded along both the primary and alternative routing paths to the sink.

The attack model We assume that once a sensor node is compromised, the adversary can retrieve all embedded security information including the secret key. Therefore, a compromised node can inject false data reports as shown in [13, 10]. We further assume the adversary knows the protocol or other security algorithms used in the network.

We assume that the adversary can attack the node association phase. Since the node association may be performed periodically at the beginning of each epoch, some nodes may have already been compromised at that time. There are up to t compromised nodes in the network.

We assume the sink always has the ability to detect a false report as shown in [13]. It is achieved by including the XOR-MAC of MACs from (t+1) sensing nodes using private keys. In this paper we therefore focus on detecting and dropping false report en-route.

4 Our Algorithm

In this section we present our algorithm for filtering false reports in multipath routing based sensor networks. We focus on enhancing node association schemes and then prove their security.

4.1 Overview

Similar to IHA, our algorithm contains five phases. The enhancements are integrated in phases 2 and 4 which will be discussed in more details next. Other phases stay unchanged.

1. The node initialization and deployment phase. Each node is loaded with a unique id and a private key before the deployment. The deployed node also sets up pairwise keys with its immediate neighbors.
2. The node association discovery phase. A node discovers its up-stream and down-stream associated nodes. Authentication keys are also generated in this phase.
3. The report endorsement phase. Each report is endorsed by (t+1) nodes within the cluster. The cluster head collects the sensing data and the (t+1) MACs, wraps them to one report, and sends the report back to the sink.
4. The en-route filtering phase. Each relay node verifies the MAC generated from its down-stream association nodes and generates *one* new MAC to be verified by its up-stream association nodes.
5. The sink verification phase. The sink node always has the ultimate ability to verify if the report is authentic using private keys of (t+1) sensing nodes.

4.2 Detailed Description

We focus our discussion on node association since the authentication largely depends on how the nodes are associated. We first present how to defend the malicious node association attack in single path routing. We then extend it to the multipath based routing network.

Enforced node association for single path routing. Ideally IHA assigns (t+1) consecutive nodes to (t+1) different authentication chains. As there are at most t compromised nodes, even if t authentication chains are broken, at least one is still well-behaved. Any node on this chain can detect and drop false reports. However in Fig. 4 with the help of the compromised node $X3$ who forges the node lists for node association, node $X2$ is successfully linked to two different authentication chains. $X2$ is linked to the $u6$-chain through direct association while it is linked to the $u4$-chain through indirect association (through $u1$). Once the malicious node association succeeds, the adversary can defeat the authentication by generating (t+1) consistent MACs from $X2$ and other $(t-1)$ compromised nodes. Since the node association is short-sighted to see past (t+1) nodes, $u4$ and $u6$ do not know that their authentication chains actually merge at $X2$.

With the above observation, the enhancement is clear – we should find a way to enforce the interleaved node association such that (t+1) authentication chains keep disjoint. A brute force approach is to remember the complete routing path from the cluster node to the sink. Clearly it is inefficient and expensive. Given there are at most t compromised nodes, we will prove it is secure by including last $(t+1)^2$ traveled nodes in the ACK message. Our enhancement works as follows,

- We include the ids of last traveled $(t+1)^2$ down-stream nodes in the ACK message, that is, each node receives a list of nodes $[u_{ij}]$ $(0 \leq i, j \leq t))$; $u_{00}, \ldots, u_{0t}$ are last traveled (t+1) nodes. (Note that the last traveled node may not be u_{00}. Instead the last traveled node is identified by the index value `Ind` as discussed next). This list is to replace the (t+1)-node list in the ACK message in IHA. These nodes form (t+1) authentication chains —

given a fixed j ($0 \le j \le t$), $u_{0j}, u_{1j}, ..., u_{tj}$ are on one authentication chain.

- An index value *Ind* with $\lceil log(t+1) \rceil$ bits is also included in the ACK message. This value is used to indicate the last traveled node and its authentication chain. IHA identifies the last node by always shifting this node to the head of the list. Instead we use a value to identify it but not to shift the list. The reason is to help node association in multipath routing case.

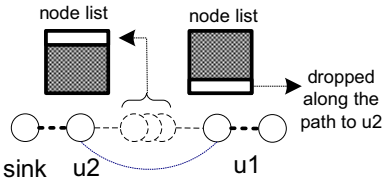
  `Ind` is initialized to zero; when a node receives the node list, it is updated as follows.

$$Ind_{new} = (Ind_{recieved} + 1) mod (t+1).$$

  After updating `Ind`, the current node is associated to the $u_{0.Ind}$ authentication chain.

- Each node also updates the node list before it sends the list to the next up-stream node. We do not shift the whole node list but instead the one column of the $[u_{ij}]$ $(0 \le i, j \le t))$ matrix. That is, the current node is set as $u_{0.Ind}$ while $u_{k.Ind}$ replaces $u_{(k+1).Ind}$ $(0 \le k \le t-1))$. Node $u_{t.Ind}$ is removed from the list.

- Instead of using the pairwise key `PairKey` between two directly associated nodes to authenticate the data reports, we generate the new authentication key `AuthKey` from the node list `[`u_{ij}`]`, the index `Ind` using the pairwise key `PairKey`. The purpose of doing so is to ensure these two nodes see common t(t+1) nodes in the list (except those removed and those added).



  After routing the list of $(t+1)^2$ ids to the up-stream associated node, its last (t+1) node ids are removed, the rest are shifted, and its first (t+1) nodes are added along the routing. To generate the consistent authentication key, we have

$$\text{at } u1: \quad AuthKey = F_{PairKey}(Ind||u_{00}||u_{01}||...||u_{(t-1)(t-1)}).$$
$$\text{at } u2: \quad AuthKey = F_{PairKey}(Ind||u_{10}||u_{11}||...||u_{tt}).$$

  Where F is an encryption or a keyed-hashing function, e.g. RC5 [10]; `PairKey` is the pairwise key shared between $u1$ and $u2$. $u1$ and $u2$ must be sharing the same `Ind` as they are on the same authentication chain.

- Before generating the new authentication key, the current node checks the received node list to ensure that the last traveled t node $u_{00}...u_{0t}$ (but not $u_{0.Ind}$) do not overlap with the nodes on its authentication chain $u_{0.Ind}...u_{t.Ind}$.

We will prove that the above enhancement ensures that any (t+1) consecutive non-compromised nodes stays on disjoint authentication chains. Now the attack in Fig. 4 cannot succeed:x $u4$ will reject the association as $X2$ appears on the $u4$-chain and it is one of the last traveled t nodes.

Node association for multipath routing. Next we discuss how to achieve secure and efficient authentication in multipath routing. We split the node association to two subtasks and want to achieve

(1) The up-stream association: to maintain similar authentication overhead as that in single path routing, each node only generates one MAC regardless of the path a report may further be forwarded to. In other words, a node shares the same authentication key with multiple up-stream nodes.

(2) The down-stream association: to ensure the authentication strengthen, we perform conservative down-stream node association assuming the path with the smallest number of hops (and authentication checks) was taken. That is, the authentication is interleaved according to the shortest path while it may be out of the interleaving order along other paths.

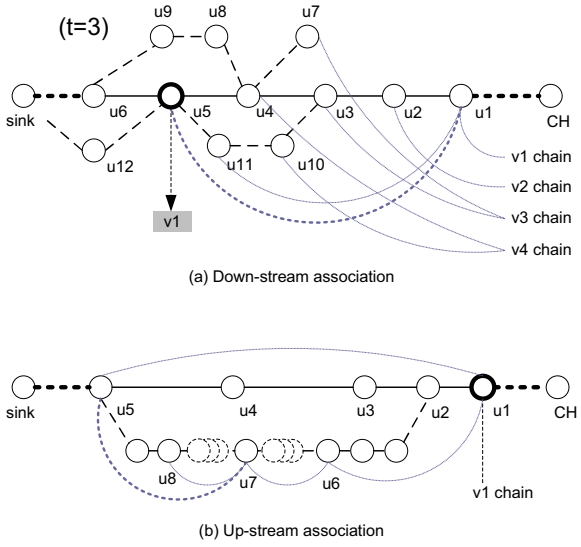


Figure 6. The conservative node association in multipath routing (t=3).

Next we use an example to illustrate how our scheme works and then present the detailed algorithm. In Fig. 6, we assume t=3 and there are four $v1, v2, v3, v4$ authentication chains. Let us first focus on the down-stream node

association at the merging node $u5$. Assuming that nodes $u1$ to $u4$ have been interleavingly associated to $v1$- to $v4$-chains respectively. node $u7$, $u10$ and $u11$ have been associated to $v3$-, $v4$-, and $v1$- chains respectively. According to two different paths $u1-u2-u3-u4-u5$ and $u1-u2-u3-u10-u11$, $u5$ can be associated either to $v1$-chain or $v2$-chain. We want to associate it to one chain to ensure security. With the conservative association policy, we associate it to the $v1$-chain, which ensures that checks on $v1$-chain are not missed along any path.

The above association leaves the authentication on the path $u1-u2-u3-u10-u11$ not strictly interleaved. More importantly, along this path, we associate $u5$ to $u1$ — a $v1$-chain node one round down-stream than $u5$'s immediate $v1$-chain node $u11$. The reason for doing so is to ensure that there are at least t nodes between any two directly associated nodes. We will use this property to prove the correctness of our scheme.

Let us then look at the up-stream association at the node $u1$ in Fig. 6(b). it has two up-stream nodes $u5$ and $u6$ both of which are four hops away. Since we only want to generate one MAC at node $u1$, nodes $u1$, $u5$, $u6$ are grouped to share one authentication key. On the other hand, a node does not have to share the same authentication key with all its down-stream nodes. That is, $u5$ may keep two authentication keys, one (with $u1$) for the packet routed along $u4-u5$ and the other one (with u7) for the packet routed along $u8-u5$.

It can be tricky with respect to the group authentication key generation. That is, assuming node $u6$ is a compromised node, it may try to associate with $u4$ as well. If it succeeds, $u6$ stays on two different chains. Thus node $u4$ has to know that $u6$ has already been on $u1$-chain in order to reject such a request. Since $u1$ does not know this beforehand, we send a SYN message which is just to find the association relationship but leave the actual key generate in processing the next ACK message. Similar to that in single path routing, the ACK message contains related node ids up to (t+1) rounds on each of the (t+1) authentication chains.

The algorithm details are as follows.

- The algorithm contains three steps. In the first step, the CH sends a SYN message from the cluster head to the sink along all braided paths. It is to decide on which authentication chain each of the relay node should be assigned to. In the second step, the sink sends a NOTIFY message to the CH. Each node gets a notification from each of its up-stream associated nodes. In the third step, the CH sends an ACK message with the related node list to generate the authentication key.
- In step 1, the SYN message contains an index `Ind` and a value `Rcnt` that counts the number of association rounds.

  If a node u_x receives one SYN message from u_y, it is assigned as the next node of u_y and updates `Ind` and `Rcnt` accordingly. If a node u_x receives more than one SYN messages, it is assigned as the next node along the path with the smallest `Rcnt`, or the smallest `Ind` if all `Rcnt` are the same. `Ind` and `Rcnt` are updated as well.

  For example in Fig. 6(a), $u5$ and $u6$ are assigned to v3- and v4- chains respectively.
- In step 2, the NOTIFY message contains those traveled up-stream nodes that have not reach their associated nodes. Each node receives the notification from all up-stream nodes and adds itself to the list to notify its down-stream associated node.

  For example in Fig. 6(b), $u1$ is notified that both $u5$ and $u6$ are to associate with it.
- In step 3, the ACK message contains the node list to generate the authentication keys. This list is organized as (t+1) authentication chains. Each chain remembers up to (t+1) rounds of its down-stream nodes and their directly associated nodes (if these nodes do not appear on the chain).

  For example in Fig. 6(b), with respect to the v1-chain/node list received by node $u4$ along the path $u1-u2-u3-u4$. Node $u6$ should have been added to the list at $u1$ since it is directly associated with $u1$ and $u1$ receives this notification in step 2.
- To generate the authentication key in step 3. All nodes in one association group, e.g. $u1$, $u5$, and $u6$ in Fig. 6(b), compute from `Rcnt`, `Ind` and the shared node ids in their received ACK messages. They share t rounds of ids on each authentication chain but the position may be shifted similar to that in single path routing. That is $AuthKey = F_S(Rcnt||Ind||Shared\_ids)$. Where S is a secret selected by $u1$ and transferred to its up-stream associated nodes using the pairwise keys.
- Before generating the authentication key, each node verifies that last traveled t nodes do not overlap with nodes on its down-stream (t+1) rounds of the authentication chain (and their directly associated ones such as $u6$ in Fig. 6(b)).

En-route message authentication. After the node association, the en-route message authentication works similar to that in single path authentication. Each report contains (t+1) MACs, each relay node picks up the one to verify according to its `Ind` and the authentication key created in the node association phase. For example in Fig. 6(a) $u5$ always picks up the third MAC to verify as it was assigned to $v3$.

After the authentication, a new MAC is generated using the authentication key for its up-stream association nodes

and replaces the old one in the report. The report is then forwarded following multiple outgoing paths.

The only difference is that, since a report may take different paths to reach a merge node, the MAC of the same index may be different along different paths. For example in Fig. 6(b) we may use either $AuthKey_{u5 \cdot u7}$ or $AuthKey_{u1 \cdot u5 \cdot u6}$. Several extra bits are therefore needed to distinguish the path. In general this overhead is small.

4.3 Security Analysis

In this section we analyze the security of our enhancements. In particular we prove that the proposed scheme can defend the malicious node association attack in both single path and multipath routing networks.

Theorem 1 *The proposed scheme can defend node association manipulation attack in single path routing sensor networks.*

Proof We will prove by contradiction. Let us assume that a false report can escape the authentication check, that is, it can route through consecutive (t+1) *non-compromised* nodes without being detected.

Let us denote the first such routing path segment as SEG_1. It contains (t+1) non-compromised nodes $v_1, v_2, ...v_{(t+1)}$. The t compromised nodes are $X_1, X_2, ..., X_t$.

Since there are (t+1) authentication chains and t compromised nodes, we can find at least one compromised node which stays on two authentication chains. Otherwise there is an authentication chain that consists of non-compromised nodes. The false report will be detected if any node on this chain receives the report.

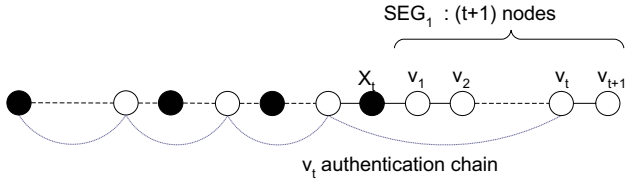


Figure 7. V_t has to meet a compromised node with t association rounds.

The node before reaching v_1 must be a compromised node, otherwise the first non-compromised (t+1)-node path segment would be from this node to v_t. Accordingly there is a compromised node between v_t and its down-stream associated node v_{dt}. Similarly we can find a compromised node between v_{dt} and its down-stream associated node, and so on. Given there are at most t compromised nodes, this process can jump at most t rounds, or a compromised node is met on the chain. If it jumps t rounds, then all t compromised nodes have been jumped and therefore it is an authentication chain without any compromised node. It can detect a false report immediately.

In other words, all (t+1) authentication chains must meet a compromised node within t rounds of association down-stream, i.e. with $(t+1)^2$ nodes down-stream. A compromised node X_1 therefore has to stay on two chains e.g. v_1 and v_2 chains. From the algorithm, at least $t(t+1)$ shared nodes received by $v1, ..., v_{(t+1)}$ cannot be compromised. Otherwise different lists are used to generate keys such that the consistent authentication key cannot be reached between associated nodes. Therefore the fact that X_1 is to associate with two chains cannot succeed.

To summarize, it is impossible to associate one compromised node with two authentication chains. Accordingly a false report cannot be routed through consecutive (t+1) *non-compromised* nodes without being detected. The node association attack is defended. ∎

Theorem 2 *The proposed scheme can defend node association manipulation attack in multipath routing based sensor networks.*

Proof To prove in multipath routing networks, we follow the similar strategy as that in the above proof.

If a false report can escape the authentication check in a reasonable large network, it has to route through a routing path segment that contains nodes from (t+1) different authentication chains. We assume the first such segment is SEG_1. It may have more than (t+1) nodes since node association is not strictly interleaved along some paths. Due to the fact that there are at most t compromised nodes, we can find at least one compromised who can generate two MACs that can be accepted into two authentication chains. This means this compromised node should share the authentication keys with some nodes on two different authentication chains.

In the node association scheme for multipath routing, a compromised node X has two choices to share an authentication key with nodes on an authentication chain. Regarding the $u5$-chain in Fig. 6(b), X can either be $u1$ which is a $u5$-chain node and on the routing from $u5$ to CH, or $u6$ which is a $u1$-chain node but not $u5$-chain node. $u6$ is directly associated with a node $u1$ on the $u5$-chain. We denote all these nodes as *related* nodes in this proof. Therefore, we can conclude that a compromised node has to be one of such related nodes on two different authentication chains.

We next prove that all compromised t nodes must be within t down-stream rounds from SEG_1. Our scheme ensures (i) between any two directly associated nodes, there are nodes from all other t authentication chains; and (ii) the ACK message includes *related* nodes from (t+1) down-stream rounds on each authentication chain. If there are less than t compromised nodes included, we can find a routing path segment which is down-stream ahead of SEG_1 and includes (t+1) non-compromised nodes. We should be focusing that segment instead of SEG_1.

Since the ACK message memorizes the *related* nodes from past (t+1) rounds on each chain, and we check node overlapping before generating the authentication key, it is impossible to let one compromised node break into two chains without being detected.

Therefore we should have at least one authentication chain which contains no compromised node. The false report will be detected and dropped immediately if any node on this chain receives the packet. As nodes are associated conservatively according to the shortest path, it is impossible to skip such an authentication chain.

In summary, we can defend the node association attack in the multipath routing. ∎

4.4 Overhead Analysis

The interleaved authentication overhead comes from the cost to setup the node association, and the cost to perform en-route report authenticate.

The overhead for node association. To achieve higher security, our node association processes three messages. The SYN message contains two values; the NOTIFY message contains a node list with O(t) node ids; the ACK messages contains a node list with $O((t+1)^2)$ node ids. In comparison, IHA has two messages each of which contains a node list of (t+1) ids. While the overhead of the node association phase in our scheme is higher, it is a one-time overhead in each epoch and amortized by multiple relayed packets.

En-route report authentication. We achieve similar en-route authentication overhead as that in IHA. Each reports contains (t+1) en-route authentication MACs. In comparison a report may have to carry $m(t+1)$ MACs if IHA is applied independently and on average each node generates m MACs.

The upper bound that a false report can travel in a multipath routing based sensor network is the same as that in IHA. Since a compromised node can forge the node list to fool its next t (but not possible (t+1)) non-compromised nodes, in the worst case, a false report may be forwarded t^2 hops from the node where the false report was generated. Since nodes are conservatively associated according to the shortest path, this upper bound equals to t^2 nodes regarding the shortest path.

Other cost includes the several bits in the report to distinguish the (t+1)-hop routing path such that a relay node can select from multiple down-stream keys. The required number of bits is in general small and can be removed if there is only one down-stream associated node.

5 Conclusions

In this paper we studied en-route false report filtering in multipath routing based sensor networks. We identified the node association manipulation attack and the association problems in multipath routing. We proposed schemes to achieve secure and efficient authentication, and analyzed their security and performance. The schemes achieve similar en-route authentication overhead and filtering upper bound as these in single path routing.

Acknowledgement

This work is partially supported by NSF CAREER grant 0447934 and NSF grant 0430021.

References

[1] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly-resilient, Energy-efficient Multipath Routing in Wireless Sensor Networks," In *ACM SIGMOBILE Mobile Computing and Communications Review*, Vol.5(4), 2001.

[2] W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An Application-Specific Protocol Architecture for Wireless Microsensor Networks," *IEEE Transactions on Wireless Communications*, vol 1:4, pages 660-670, 2002.

[3] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed Diffusion: a Scalable and Robust Communication in Wireless Sensor Networks," In *5th IEEE/ACM Mobicom*, pages 174-185, 1999.

[4] C. Karlof and D. Wagner, "Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures," in *IEEE International workshop on Sensor Network Protocols and Applications*, pages 113-127, 2003.

[5] J.N. Al-Karaki, and A.E. Kamal, "Routing Techniques in Wireless Sensor Networks: A Survey," in *IEEE Wireless Communications*.

[6] S. Dulman, T. Nieberg, J. Wu, and P. Havinga, "Trade-off between Traffic Overhead and Reliability in Multipath Routing for Wireless Sensor Networks," In *WCNC workshop*, 2003.

[7] V. Kawadia and P. R. Kumar, "Power Control and Clustering in Ad Hoc Networks," In *INFOCOM*, 2003.

[8] Y. Xu, J. Heidemann, D. Estrin, "Geography-informed Energy Conservation for Ad Hoc Routing," in *ACM MOBICOM*, 2001.

[9] H. Yang, F. Ye, Y. Yuan, S. Lu and W. Arbaugh, "Toward Resilient Security in Wireless Sensor Networks," In *ACM MOBIHOC'05*, 2005.

[10] F. Ye, H. Luo, S. Lu and L. Zhang, "Statistical En-route Detection and Filtering of Injected False Data in Sensor Networks," In *IEEE INFOCOM 2004*, 2004.

[11] F. Ye, G. Zhong, S. Lu, and L. Zhang, "GRAdient broadcast: A robust data delivery protocol for large scale sensor networks," In *ACM Wireless Netw. (WINET)*, vol. 11, no. 2, Mar. 2005.

[12] W. Zhang and G. Cao, "Group Rekeying for Filtering False Data in Sensor Networks: A Predistribution and Local Collaboration-Based Approach," In *INFOCOM*, 2005.

[13] S. Zhu, S. Setia, S. Jajodia, P. Ning, "An Interleaved Hop-by-Hop Authentication Scheme for Filtering of Injected False Data in Sensor Networks," In *Proceedings of IEEE Symposium on Security and Privacy*, Oakland, California, May 2004.