Available online at www.sciencedirect.com
ScienceDirect
J. Parallel Distrib. Comput. 67 (2007) 1218–1230
Journal of Parallel and Distributed Computing
www.elsevier.com/locate/jpdc

CHEMAS: Identify suspect nodes in selective forwarding attacks☆

Bin Xiao[a,*], Bo Yu[a,b], Chuanshan Gao[c]

[a]*Department of Computing, Hong Kong Polytechnic University, Hong Kong*
[b]*Department of Electrical and Computer Engineering, Wayne State University, USA*
[c]*Computer Science and Engineering Department, Fudan University, PR China*

Received 2 September 2006; received in revised form 16 March 2007; accepted 30 April 2007
Available online 3 June 2007

Abstract

Selective forwarding attacks may corrupt some mission-critical applications such as military surveillance and forest fire monitoring in wireless sensor networks. In such attacks, most of the time malicious nodes behave like normal nodes but will from time to time selectively drop sensitive packets, such as a packet reporting the movement of the opposing forces, and thereby make it harder to detect their malicious nature. In this paper, we propose CHEMAS (CHEckpoint-based Multi-hop Acknowledgement Scheme), a lightweight security scheme for detecting selective forwarding attacks. Our scheme can randomly select part of intermediate nodes along a forwarding path as checkpoint nodes which are responsible for generating acknowledgements for each packet received. The strategy of random-checkpoint-selection significantly increases the resilience against attacks because it prevents a proportion of the sensor nodes from becoming the targets of attempts to compromise them. In our scheme, each intermediate node in a forwarding path, if it does not receive enough acknowledgements from the downstream checkpoint nodes, has the potential to detect abnormal packet loss and identify suspect nodes. We explore the feasibility of our detection scheme using both theoretical analysis and simulations. The simulation results show that our scheme can achieve a high detection rate, even in harsh radio conditions. The communication overhead incurred by our scheme is also within reasonable bounds.
© 2007 Elsevier Inc. All rights reserved.

Keywords: Wireless sensor networks; Intrusion detection; Selective forwarding attacks

1. Introduction

Wireless sensor networks (WSNs) are ideal candidates for monitoring environments in a wide variety of applications such as military surveillance and forest fire monitoring [14]. In such a network, a large number of sensor nodes are deployed over a vast terrain to detect events of interest (e.g., enemy vehicles, outbreaks of forest fires), and to deliver data reports to the base station over multi-hop wireless paths. The node-patterned deployment of WSNs, however, can be the focus of certain types of malicious attack. One such strategy is the selective forwarding attack, first proposed by Karlof [6]. Fig. 1 shows an example sensor network under selective forwarding attacks. As shown in the figure, two compromised nodes selectively drops sensitive packets, for example, a packet reporting the enemy tank movements. The dropping rate should be higher than the normal packet loss rate (i.e., due to the poor channel condition) to ensure no sensitive packet pass. This kind of attack is typically most effective when the attacking nodes are explicitly included on the path of a data flow. They can corrupt a number of existing routing protocols such as TinyOS beaconing, directed diffusion [5], GPSR [7], GEAR [17], and clustered based protocols, especially when they are used in combination with other attacks such as wormhole and sinkhole attacks.

The adversary may incur abnormal packet loss in two ways, from inside the network via maliciously dropping packets going through compromised nodes or from outside the network by jamming the communication channels between uncompromised nodes, as shown in Fig. 1. Usually, adversaries prefer inside attacks because they put the adversary in a position to know more about passing packets, thereby enabling them to selectively drop sensitive packets. In this paper, we also mainly focus on selective forwarding attacks from inside compromised nodes.

☆ This work is supported by HK RGC CERG B-Q02S and POLYU A-PA2F.
* Corresponding author.
E-mail address: csbxiao@comp.polyu.edu.hk (B. Xiao).

0743-7315/$ - see front matter © 2007 Elsevier Inc. All rights reserved.
doi:10.1016/j.jpdc.2007.04.014

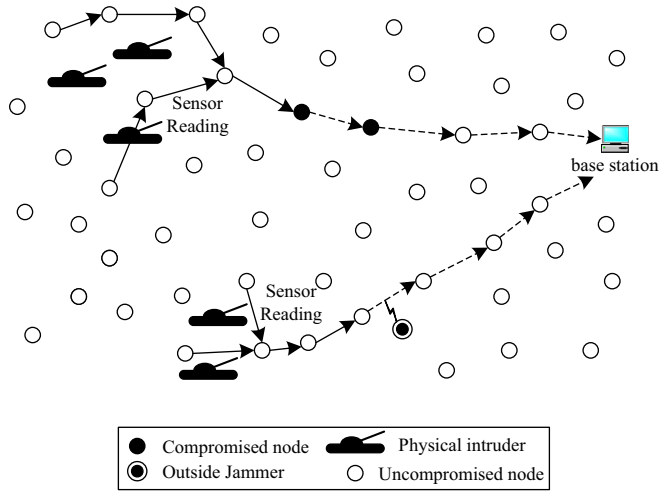


Fig. 1. An example sensor network under selective forwarding attacks.

One possible approach that can be used to decrease the impact of selective forwarding is to use a multipath forwarding technique, which is based on packet delivery redundancy. However, multipath forwarding suffers from several drawbacks. First, communication overheads increase dramatically as the number of paths increase. Second, multiple paths ultimately join up in the area neighboring the base station, so if nodes around the base stations are compromised, selective forwarding is still applicable. Finally, the multipath forwarding shows limited security resilience. To compromise the system, an adversary merely needs to ensure the presence of one compromised no de in each path.

In this paper, we propose CHEMAS (CHEckpoint-based Multi-hop Acknowledgement Scheme), a lightweight security scheme that detects selective forwarding attacks by using a checkpoint-based acknowledgement technique. This paper seeks identification and localization of suspect nodes by a random-checkpoint-selection strategy. With this strategy, parts of intermediate nodes along a forwarding path can be randomly selected as checkpoint nodes, which are responsible for acknowledgement for each packet safely delivered to them. This random-checkpoint-selection strategy can significantly increase the system resilience by preventing sensor nodes from becoming the targets of attempts to compromise them. In our scheme, each intermediate node in a forwarding path has the potential to detect abnormal packet loss and identify suspect nodes if it does not receive enough acknowledgements from the downstream checkpoint nodes. The source nodes can collect alert information containing both suspect nodes' ID and position information from intermediate nodes. This source-side detection mechanism is so advantageous that even when the base station is deafened by surrounding malicious nodes, the source nodes remain capable of making decisions and responding. In our simulations, the communication overhead of our scheme is less than 1.5 times of that in a system that does not incorporate our scheme, and the detection accuracy is over 95% even when the channel error rate is harsh 15%.

The remainder of this paper is organized as follows. Section 2 introduces some preliminary work for our scheme, containing several assumptions, the location-binding node ID technique, as well as the suggestion of whole defense process. Section 3 presents our detection scheme in detail based on checkpoint-based multi-hop acknowledgement. In Section 4, we discuss the identification of suspect nodes. Section 5 first proposes several evaluation metrics for our detection scheme and then shows the simulation results for these metrics. In Section 6, we discuss several other potential approaches to improve the detection. Finally, Section 7 introduces the related work and Section 8 concludes our work.

2. Preliminaries

In this section, we make several assumptions, propose a location-binding node ID technique and then suggest the entire process for defending against selective forwarding attacks in a detection-based manner.

2.1. Assumptions

We make five assumptions in applying the proposed detection scheme in a mission-critical application of WSNs such as military reconnaissance. First, we assume that during the deployment phase each sensor can acquire its geographical position and loosely synchronize its time with the base station. Second, we assume that the adversary cannot successfully compromise a node during the short deployment phase. This assumption has proved appropriate in some existing work [14]. Third, we assume that in order to avoid arousing suspicion, malicious nodes selectively drop a small proportion of all packets passing, that is, they will not drop all packets. Fourth, we assume that an authenticated broadcast protocol such as μTESLA [10] has been implemented in each node. Finally, we assume that routing and transport protocols such as directed diffusion [5] and PSFQ [12] have also been implemented in each node. Our scheme can function over these protocols when the network topology is relatively static during the attack period.

Although the routing layer of WSNs may be threatened by a variety of attacks, here we are considering only selective forwarding attacks.

2.2. Location-binding node ID

To help to localize suspect nodes, we propose a location-binding node ID technique. This technique allows any two sensor nodes in a network to easily establish a session key as long as they know each other' node ID. Their geographical location can also be derived from their node IDs.

Note that our design goal is to identify and exclude the malicious nodes. Hence it is important to compute the positions of malicious nodes after detecting selective forwarding attacks. Since sensor nodes are static after deployment in most sensor applications, especially in applications of mass deployment, it is unnecessary to allocate each sensor node a traditional unique ID. We can let each sensor node have a location-binding ID so

that as long as a node ID is known, the position of the node can be easily computed. However, an important issue is how to prevent the malicious nodes from fabricating their IDs and positions. We take advantage of bivariate polynomials to address this issue.

Our location-binding node ID technique consists of two steps. First, before deployment, the key server generates a symmetric bivariate d-degree polynomial $f(u, v) = \sum_{i,j=0}^{d} a_{ij} u^i v^j$ over a finite field F_q, where q is a prime number that is large enough to accommodate a cryptographic key. Here, a field is referred to as any set of elements, and a finite field as a field with a prime number of elements. A polynomial $f(u, v)$ is said to be symmetric if $f(u, v) = f(v, u)$. Then the key server loads each node with the polynomial.

Second, as soon as the nodes are deployed into the target field, they begin to acquire their geographical position through secure positioning algorithms such as [1]. Each node binds their ID with their geographical position: $ID = x\|y$, where $\|$ denotes concatenation and (x, y) denotes the geographical coordinates. Then the node regenerates a polynomial for pairwise key establishment: $g(v) = f(ID, v)$, and erases $f(u, v)$.

We take the following example to show how our location-binding node ID technique works. Suppose that node a owns $g_a(v) = f(a, v)$, node b owns $g_b(v) = f(b, v)$, and node a wants to send a packet to node b. First, node a calculates $k_1 = g_a(b)$ as the pairwise key and encrypts the packet using k_1. Then node a sends the packet, along with its ID, a, to node b. When node b receives the packet, it calculates $k_2 = g_b(a)$, ($k_2 = k_1 = f(a, b) = f(b, a)$), as the decryption key and decrypts the packet. If the decryption is successful, node b believes the packet comes from the authentic node a, and node a's location (x, y) can be easily derived from node a's id ($a = x\|y$).

Our location-binding node ID technique is simple and efficient. It has several desirable features. First, location information is bound to a node ID, thereby incurring no extra memory and communication overhead. Second, node ID and location information can be authenticated using polynomial-based keys. It is infeasible for the adversaries to fabricate nodes location information, even when the adversaries have compromised a number of nodes and obtained all the secret information stored in the nodes' memory. The adversary also may physically move the compromised nodes in order to hide the nodes' real position, but the possible range of movement is quite limited, because if a compromised node is moved to a region distant from its original position, the uncompromised nodes in the distanced region will believe the compromised node does not belong in that region and will refuse to communicate with it. Finally, in terms of security strength, our location-binding technique is quite resilient. The adversary may try to regenerate the original bivariate polynomial $f(u, v)$ to crack the whole network but after the original bivariate polynomial $f(u, v)$ is erased from each nodes' memory during the deployment phase, it will be infeasible for the adversaries to regenerate $f(u, v)$, as long as no more than k nodes are compromised. When used in key establishment, symmetric bivariate polynomials have proved unconditionally secure as long as no more than k nodes are compromised [19].

2.3. Entire defensive process

We suggest that the entire defensive process should consist of three phases, *a deployment phase*, *an intrusion detection phase*, and *a decision and response phase*.

In the deployment phase, each node, after deployed into the target field, should finish several preliminary operations, which are required by protocols implemented in each node. For our scheme, each node need to find neighboring nodes which might be multiple hops away, and then exchange the first key in its key chain with its neighbors, which is required by the μTESLA protocol [10].

In the intrusion detection phase, the network performs its main task such as military surveillance, while the intrusion detection mechanism monitors abnormal packet loss during packet delivery. In this paper, we mainly focus on the intrusion detection phase.

The network comes into the decision and response phase, when enough evidence has been collected. We can make decisions about which nodes are malicious nodes by using statistic techniques or other complicated IDS algorithms. Then, an alarm packet can be generated and broadcasted, finally excluding the malicious nodes from the whole network.

3. CHEMAS: CHEckpoint-based Multi-hop Acknowledgement Scheme

In this section, we introduce our detection scheme in detail. A simple approach to detect packet loss is *acknowledgement*. Traditional transport layer protocols such as PSFQ [12,11] use hop-by-hop acknowledgement to ensure reliable delivery of packets. However, such protocols are not designed to deal with malicious attacks. Inspired by traditional hop-by-hop acknowledgement, we believe that we can use a multi-hop acknowledgement technique to verify whether intermediate nodes faithfully forward each packet passing by.

3.1. Definitions of packets

We first introduce three packets which will be used in our detection scheme. The suggested packet format for each packet is provided in Fig. 2.

Event packets are generated at the source nodes when a special event, e.g., a tank movement noise, is detected, or in response to a query from a base station. After an event packet is generated, it is forwarded hop-by-hop from the source node to the base station. The suggested packet fields are shown in Fig. 2(a), containing *DstID*, *SrcID*, *Packet_ID*, *Payload*, and *Checkpoint_Seed*. The first four elements are ordinary fields as required by routing protocols, whereas *Checkpoint_Seed* is a number, deciding which intermediate nodes along a forwarding path are selected as checkpoint nodes.

ACK packets are generated at checkpoint nodes in a forwarding path. When a checkpoint node receives an event packet, it generates an ACK packet for the event packet and then delivers it to the upstream nodes. The ACK packet follows the same path as traversed by the previous event packet but in the

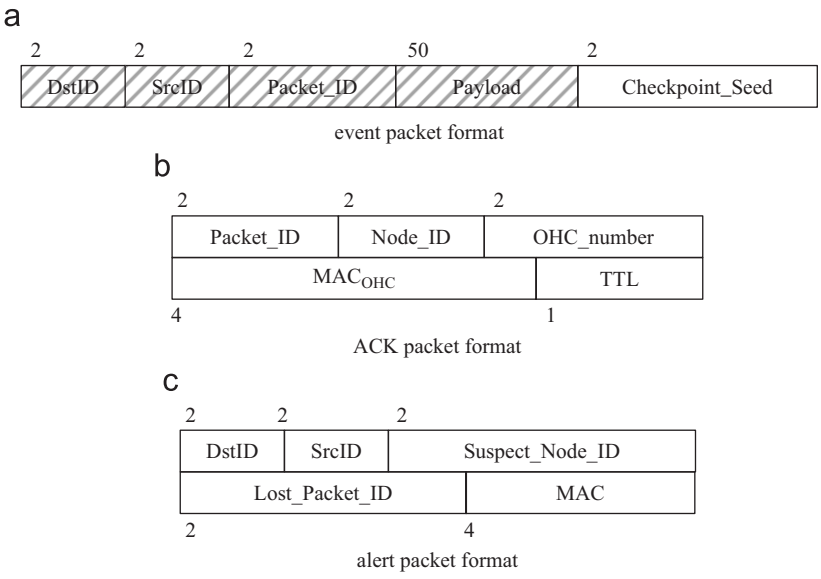


Fig. 2. Packet formats.

opposite direction. An ACK packet contains *Packet_ID*, *Node_ID*, *OHC_number*, MAC_{OHC}, and *TTL*. *Packet_ID* refers to the previous report packet id which this ACK packet acknowledges. *Node_ID* refers to the current node which generates the ACK packet. *OHC_number* and MAC_{OHC} are used in the authenticated broadcast protocol such as [10]. *TTL* decides the number of checkpoint nodes an ACK packet can traverse before being dropped. Initially, this field is set to 0 when an ACK packet is generated. Each time the packet is delivered to the next checkpoint, this field is increased by one.

Alert packets are generated at intermediate nodes when suspect nodes are detected. Once generated, alert packets will be sent to the source node or the base station through multiple hops. An alert packet contains *DstID*, *SrcID*, *Suspect_Node_ID*, *Lost_Packet_ID*, and *MAC*. *SrcID* refers to the node which generates the alarm packet, while *DstID* refers to the destination node ID. *Suspect_Node_ID* refers to the suspect node being prosecuted in an alert packet. Finally, *MAC* endorses the entire ACK packet, making it difficult for malicious nodes to fabricate or modify alert packets.The *MAC* can be generated by using the location-binding ID technique introduced in Section 2.2.

3.2. Detection scheme

In this subsection, we discuss how our scheme works during the detection phase. Unlike traditional hop-by-hop acknowledgement in transport layer protocols of sensor networks, our scheme is based on checkpoint-by-checkpoint acknowledgement.

The basic idea of our scheme is as follows. Part of the intermediate nodes along the forwarding path are selected as checkpoint nodes, as shown in Fig. 3. The path then is divided into several segments by these checkpoint nodes. In other words, a *segment* consists of all intermediate nodes between two consecutive checkpoint nodes in a forwarding path. When the source node detects a special event, e.g., tank movement noise, an event packet is generated. The event packet will be forwarded hop-by-hop toward the base station, and each intermediate node saves the event packet in its cache after forwarding it to the next downstream node. When a checkpoint node receives an event packet from an upstream neighbor, it generates an ACK packet, which is signed with the next OHC number, and then sends the ACK packet back to the upstream neighbor. The ACK packet is transferred toward the source node along the same but reversed path as the previous event packet. It traverses at least two segments before being dropped by an upstream checkpoint. Thus all the nodes in these two segments know that the previous event has safely arrived at the downstream checkpoints. If an intermediate node cannot receive ACK packets from downstream, it will generate an alert packet, specifying the next downstream-neighboring node as the suspect node.

To show how our scheme works, we take Fig. 3 as an example. In this example, node $u4$, $u6$, and $u9$ are selected as checkpoint nodes, and ACK packets are supposed to traverse two segments before being dropped. First, the source node generates an event packet and sends it to node $u1$, and then the packet is forwarded hop-by-hop toward the base station. Each intermediate node saves the packet in its cache, forwards it to the next downstream node, and then waits for ACK packets from the downstream. When node $u4$ receives the event packet, node $u4$ generates an ACK packet. It does this because of its checkpoint identity. The packet traverses node $u3$, $u2$, and $u1$ and is finally dropped at the source node. Thus nodes $u3$, $u2$, $u1$, and the source node know that the previous event packet has already arrived at the next checkpoint $u4$. Thus nodes $u3$, $u2$, $u1$, and the source node continue waiting for the ACK packet from checkpoint $u6$. In this way, each intermediate node can verify whether the event packet has safely arrived at the downstream checkpoints.

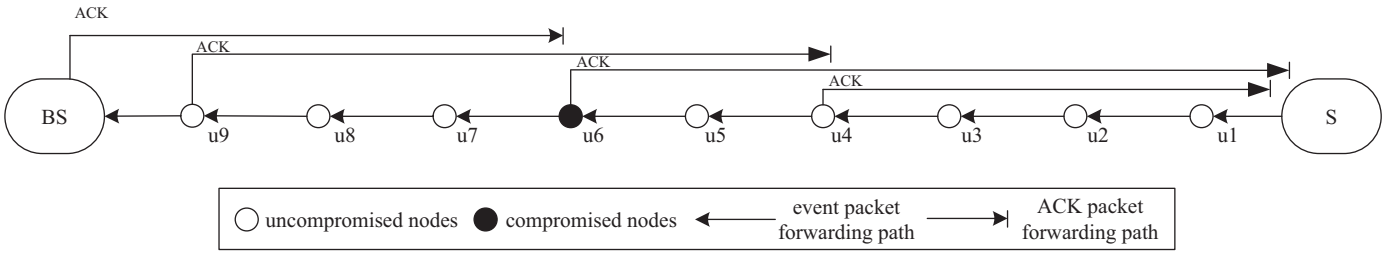


Fig. 3. An example of multi-hop acknowledgement. Node $u4$, $u6$, and $u9$ are selected as checkpoint nodes in this example.

Next we consider what might happen in the example just given if malicious nodes were included. Suppose that node $u6$ is compromised during the event forwarding phase. Node $u6$ wants to selectively drop some of the event packets going through it, but any packet losses would be detected by nodes $u5$ and $u4$. If node $u5$ does not receive an ACK packet from node $u9$, $u5$ will generate an alert packet which will be delivered to the source node through multiple hops. This alert packet will specify node $u6$, the next downstream neighbor of $u5$, as the suspect node. When the source node collects enough evidence, it will send out an alarm packet, which will finally exclude node $u6$ from the network.

In the next two subsections, we will discuss two important issues in our checkpoint-based multi-hop acknowledgement.

3.3. Checkpoint selection

In our checkpoint-based acknowledgement scheme, one important issue is how the checkpoints are selected.

One naive approach is to generate a fixed list of checkpoint nodes before the source node sends out the first event packet. However, this approach is infeasible, because checkpoint nodes are responsible for generating ACK packets, and if the checkpoint nodes are compromised, the adversary may crack the network by fabricating ACK packets and without being detected. Therefore, it is important for intermediate nodes to share the probability of being selected as checkpoint nodes as well as the risk of being compromised.

We propose a random-checkpoints-selection algorithm, in which a random list of checkpoint nodes is generated by the source node for each event packet. The source node generates a random number as a seed for each event packet, and this seed determines the members of the checkpoint list.

This checkpoint selection algorithm has two main steps: *intermediate node bootstrapping*, and *random-checkpoint-based acknowledgement*.

(1) *Intermediate node bootstrapping*: Before deployment, each node is loaded with two functions: an one-way hash function $F_{ID}(x)$ and a map function $f_p(y)$, where ID is the sensor node ID and p is a pre-defined probability. The function $f_p(y)$ has the following attributes: $Range(f_p) \rightarrow \{0, 1\}$, and when $y \in Range(F_{ID})$, the inputs of f_p maps to 1 at probability p and 0 at probability $(1 - p)$.

(2) *Random-checkpoint-based acknowledgement*: The source node generates a random number, r, for the *Checkpoint_Seed* field in each event packet. Then the event packet is forwarded hop-by-hop toward the base station. When each intermediate node receives the event packet, the node checks whether $f_p(F_{ID}(r))$ is equal to 1. If so, the node knows that it is required to be a checkpoint node. Then an ACK packet is generated and forwarded upstream.

This random-checkpoint-selection technique ensures that p percent of intermediate nodes along a forwarding path can be randomly selected as checkpoint nodes. When an upstream node receives the ACK packet, it performs two tests on the packet. The first test is to verify that the ACK packet comes from the right checkpoint node. The first test is passed only if $f_p(F_{ID'}(r))$ is equal to 1, where ID' is $sensor\_id$ in the ACK packet, namely, the ID of sensor node from which the ACK packet is dispatched. The second test is to verify that the downstream checkpoint node does receive the previous event packet without being modified. This test can also be finished via the MAC mechanism for ACK packets, which is introduced in Section 3.1. If both these two tests are passed, the upstream node accepts that the previous event has safely arrived at the checkpoint node which dispatched the ACK packet.

The advantage of our random-checkpoint-selection algorithm is twofold. First, the possibility of either being selected as a checkpoint or being compromised is the same for every node. Second, as the selection for each event packet is independent and randomized, it is difficult for an adversary to know the list of checkpoints for the next event packet.

3.4. k-Covered Acknowledgement

In this section, we discuss how many segments an ACK packet should traverse before being dropped, or more specifically, the acknowledgement process that we call k-covered acknowledgement, wherein an ACK packet traverses k segments before being dropped. A checkpoint node can make a decision about whether to drop the packet by checking the *TTL* field in the ACK packet. If $TTL = k$, then the ACK packet is to be dropped.

Fig. 4 provides three examples of k-covered acknowledgement. From Fig. 4(a)–(c), it is easy to see that if an ACK packet traverses k segments before being dropped, each intermediate node can receive k ACK packets from the downstream checkpoint nodes. As shown in Fig. 4(a)–(c), respectively, anywhere in the forwarding path is covered with one, two, or three

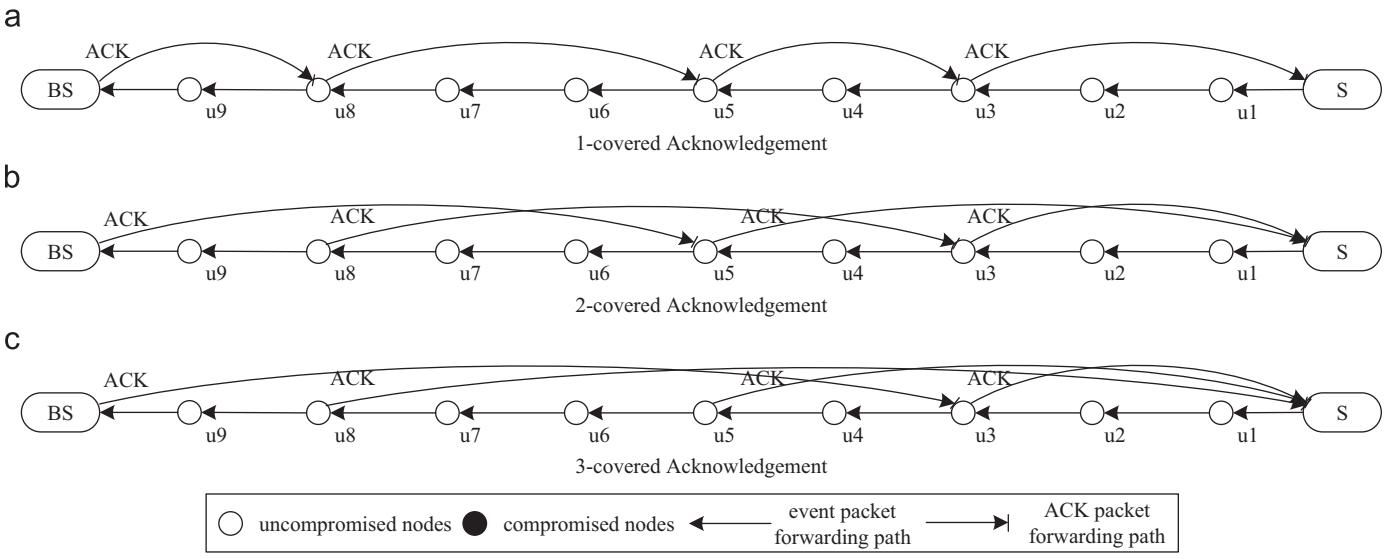


Fig. 4. Examples of k-covered acknowledgement.

ACK packet(s). However, the segments near the base station are an exception, e.g., the last segment near the base station in Fig. 4(b), and the last two segments near the base station in Fig. 4(c). The nodes in these segments may receive only one ACK packet from the base station but not k ACK packets, because the event packet has safely arrived at the base station, and no more ACK packets need to be dispatched.

The parameter k provides a tradeoff between security resilience and communication overhead. The greater k is, the more resilient is security, but communication overhead will also increase. It is easy to see that when k malicious nodes happen to be selected as k consecutive checkpoint nodes for a event packet, they can drop event packets at will and without fear of detection. For example, in Fig. 4(b), suppose that node $u5$ and $u8$ are compromised and happen to be selected as two consecutive checkpoint nodes for a event packet. When node $u5$ receives the event packet, it generates two ACK using the secret information from its own memory and that of node $u8$, respectively. We assume that the malicious nodes can cooperate with each other. In other words, they share secret information with each other. When node $u4$ and the other upstream nodes receive the two ACK packets, they believe that the packet has safely arrived at checkpoint nodes $u5$ and $u8$. However, node $u5$ simply drops the event packet instead of forwarding it anymore. This malicious dropping will not be detected by the upstream nodes. As the same time, please note that in the above example, node $u5$ and $u8$ can drop only one event packet, because the checkpoint list varies for each event packet, meaning that next time they may not be selected as checkpoint nodes.

3.5. Attack analysis

In this subsection, we discuss several potential responses that the adversary might take with the goal of cracking our detection scheme.

A compromised node might alter the *Checkpoint_Seed* field in an event packet with the goal of allowing more compromised nodes downstream to be selected as checkpoint nodes. It is worth noting that in our scheme, the authentication of *Checkpoint_Seed* fields is not ensured by any MAC mechanisms. However, the adversary does not benefit from doing so. For instance, in Fig. 3, suppose that nodes $u6$ and $u7$ are compromised, and that node $u6$ changes the *Checkpoint_Seed* field in an event packet so that according to the changed *Checkpoint_Seed*, $u7$ is selected as a checkpoint node. Note that node $u5$ owns the authentic *Checkpoint_Seed*, so when node $u5$ receives an ACK packet from node $u7$, it can detect that $u7$ is not specified as a checkpoint node by the original *Checkpoint_Seed*. Then node $u5$ simply discards this fabricated ACK packet, meaning that node $u5$ will later generate an alert packet because of not receiving enough ACK packets. Therefore, it is very possible that an intermediate compromised node can change the *Checkpoint_Seed* field (increase it or decrease it). However, it is no good doing so, because upstream nodes still hold the right *Checkpoint_Seed* and will prosecute the compromised node later. Finally, please note that the defense against modifying the content of the *Payload* field and other fields is not covered in this paper as we assume in Section 2.1.

As for ACK packets, a compromised node might fabricate false ACK packets or maliciously drop ACK packets from normal checkpoint nodes. First, a compromised node cannot fabricate ACK packets from downstream checkpoint nodes, as long as the downstream checkpoint nodes have not been compromised. This is guaranteed by the MAC mechanism for ACK packets as we introduce in Section 3.1. Second, the adversary does not benefit from malicious dropping ACK packets. This behavior prevent the upstream neighbor node from receiving enough ACK packets, which in turn makes it easier to detect the compromised nodes.

A compromised node might also fabricate alert packets maliciously prosecuting innocent normal nodes. A compromised

node may even want to prosecute all the nodes in the network, making the system totally confused. In our scheme, we have a limit that a node can prosecute only the immediately neighboring nodes in an alert packet. The limit is guaranteed by using the location-binding ID technique introduced in Section 2.2. After receiving an alert packet, the source node can verify whether the prosecuting node and the prosecuted node in an alert packet are geographically adjacent to each other. If not, the source node simply ignores the alert packet. It is true that compromised nodes can maliciously prosecute innocent nodes. Actually, it is difficult or impossible to prevent a compromise-yet-undetected node from prosecuting innocent nodes, as long as it is undetected. However, the impact of malicious prosecution is quite limited and localized, which implies that the adversary's attempt to prosecute all the nodes in a network would be fruitless.

3.6. Detection probability analysis

In this subsection, we provide a theoretical analysis of the detection probability of our scheme. For the sake of simplicity, we assume that the network is operating under ideal radio conditions, and that, consequently, packet loss must be the result of malicious dropping.

We suppose that there are n sensor nodes in a forwarding path, m of which are malicious nodes. The source node and the base station are not included in the n nodes. For simplicity, we assume that the malicious nodes are distributed in such an interleaved fashion that there are q non-malicious nodes between every two malicious nodes. k-covered acknowledgement is adopted. α is the percentage of nodes which are randomly selected as checkpoint nodes.

Our goal is to calculate the probability that malicious dropping can be detected, $P_{\text{detection}}$, which equals to the probability that less than k consecutive checkpoint nodes are malicious nodes, $P_{\text{less\_}k}$. Please note that the selection of k but inconsecutive malicious nodes as checkpoint nodes is not sufficient for the adversary to generate k fabricated ACK packets. For example, in Fig. 3, where $k = 2$, suppose that checkpoint node $u4$ and $u9$ are malicious nodes, but node $u6$ is an non-malicious node. Node $u4$ attempts to generate two ACK packets and sends them to node $u3$, but it is unable to fabricate an ACK packet from node $u6$. Thus, in this case, node $u4$ is sure to be detected if it drops an event packet.

We can compute the detection probability $P_{\text{detection}}$ as

$$P_{\text{detection}} = P_{\text{less\_}k} = 1 - P_{k\_\text{mali}},$$

where P_{k_mali} represents the probability that at least k malicious nodes are selected as consecutive checkpoints. Note that there may be malicious nodes interleaved between two consecutive malicious checkpoint nodes, but they are not selected as checkpoint nodes. P_{k_mali} can further be computed as

$$P_{k\_\text{mali}} = \sum_{i=k}^{m} P_{\text{exact\_}i}(i).$$

$P_{\text{exact\_}i}(i)$ refers to the probability that exactly i $(k \leqslant i \leqslant m)$ malicious nodes are selected as checkpoint nodes. Suppose that there are $(j-2)$ $(i \leqslant j \leqslant m)$ malicious nodes between the first malicious checkpoint node and the last malicious checkpoint node in the path, which also means that some of the nodes are not selected as checkpoint nodes. Thus, the probability that exactly i $(k \leqslant i \leqslant m)$ malicious nodes are selected as checkpoint nodes can be given by

$$P_{\text{exact\_}i}(i) = \sum_{j=i}^{m} P_{\text{exact\_}i\_j}(i, j).$$

$P_{\text{exact\_}i\_j}(i, j)$ represents the probability that exactly i of m malicious nodes are selected as checkpoint nodes and there are exactly $(j-2)$ malicious nodes between the first malicious checkpoint node and the last malicious checkpoint node. Then $P_{\text{exact\_}i\_j}(i, j)$ can be computed as

$$P_{\text{exact\_}i\_j}(i, j) = \frac{(m-j+1)\cdot\binom{j-2}{i-2}\cdot\binom{n-m-(j-1)\cdot q}{\alpha\cdot n-i}}{\binom{n}{\alpha\cdot n}}.$$

Finally, the detection probability $P_{\text{detection}}$ can be given by

$$P_{\text{detection}} = 1 - \sum_{i=k}^{m}\sum_{j=i}^{m} \frac{(m-j+1)\cdot\binom{j-2}{i-2}\cdot\binom{n-m-(j-1)\cdot q}{\alpha\cdot n-i}}{\binom{n}{\alpha\cdot n}}.$$

Fig. 5 illustrates the impact of q, k, α and m on detection probability based on the results of the preceding analysis. We assume that the forwarding path consists of 100 intermediate nodes $(n = 100)$, of which m are compromised, and that there are q uncompromised nodes interleaved between every two compromised nodes. First, we investigate the impact of deployment pattern of malicious nodes on detection probability. Fig. 5(a) shows that the curve with $q = 0$ will produce much lower detection probability than the curves with $q = 2$ and 4, which means that to avoid being detected, the adversaries would prefer to compromise consecutive nodes rather than non-consecutive nodes along a forwarding path. Hence, to analyze the performance of our scheme in the worst case, for Fig. 5(b) and (c), we suppose a scenario where consecutive nodes are compromised, namely, $q = 0$. Next, we examine the impact of k-covered acknowledgement on detection probability, as shown in Fig. 5(b). When $k \geqslant 2$, our scheme shows better performance even when 10 consecutive nodes are compromised, having a detection probability that is still above 75%. However, when $k = 1$, the detection probability is quite low. Intuitively, given $k = 1$, as long as one of the m malicious nodes is selected as a checkpoint node, which is a high probability event, the adversary can maliciously drop packets at will without being detected. Finally, Fig. 5(c) shows the impact of α on detection probability. We find that when k and m are fixed, a smaller α leads to a larger detection probability. The selection of fewer checkpoints reduces the probability of selecting malicious nodes and this in turn contributes to maintaining a high detection probability.

Analysis of the preceding results allows us to draw three conclusions. First, to avoid being detected, the adversaries prefer to compromise consecutive nodes rather than interleaved nodes

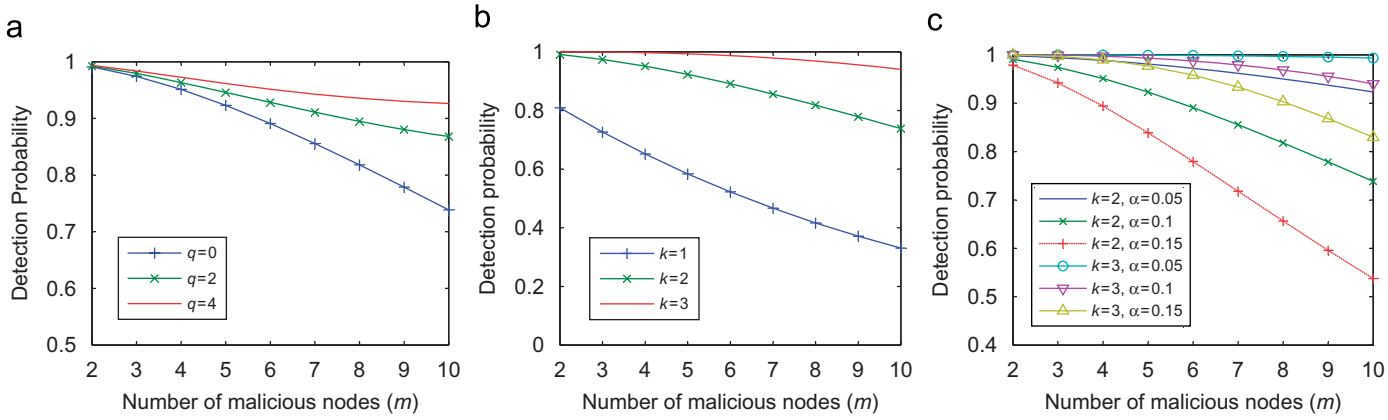


Fig. 5. Impact of q, k, α and m on detection probability: (a) impact of q and m, given $k = 1$, $\alpha = 0.1$, and $n = 100$; (b) impact of k and m, given $q = 0$, $\alpha = 0.1$, and $n = 100$; (c) impact of α and m, given $q = 0$ and $n = 100$.

along a forwarding path. Second, $k \geqslant 2$ is the minimal security requirement for our scheme. Finally, under perfect channel conditions, the selection of fewer checkpoint nodes produce better detection performance.

4. Identification of suspect nodes

In this section, we discuss the identification of suspect nodes. Since this paper does not focus on the decision and response phase introduced in Section 2.3, our purpose here is to discuss the policy which identifies suspect nodes in terms of a single alert packet reported by an intermediate node. This policy allows the source node, after receiving enough evidence, to make some more tactical decisions based on this policy, such as reducing the traffic going through the suspect nodes.

We take Fig. 6 as an example showing the identification of suspect nodes. First, we suppose that compromised nodes do not generate alert packets with the goal of maliciously prosecuting other normal nodes. Suppose that node $u4$ is a compromised node. Node $u3$ generates an alert packet prosecuting node $u4$ as a suspect node. In this case, the prosecuted node is a compromised node. Next, we suppose that compromised nodes are smart and can generate alert packets in order to maliciously prosecute innocent neighboring nodes. For instance, node $u4$ prosecutes node $u5$ by generating an alert packet. In this case, the prosecuting node is a compromised node, while the prosecuted node in the packet is innocent. As long as it remains undetected, it is a simple matter for a compromised-yet-undetected node to maliciously prosecute an innocent neighboring node. It is quite difficult, however, to distinguish between the compromised and the innocent. Consequently, in practice, in terms of a single alert packet, we should tag both the prosecuting node and the prosecuted node in the alert packet as suspect nodes.

After collecting enough evidence to make decisions, the source nodes can derive the geographical position of suspect nodes based on the location-binding ID technique introduced in Section 2.2. As future work, we are going to develop more considerate decision and response mechanisms for the source nodes or the base station, based on the information

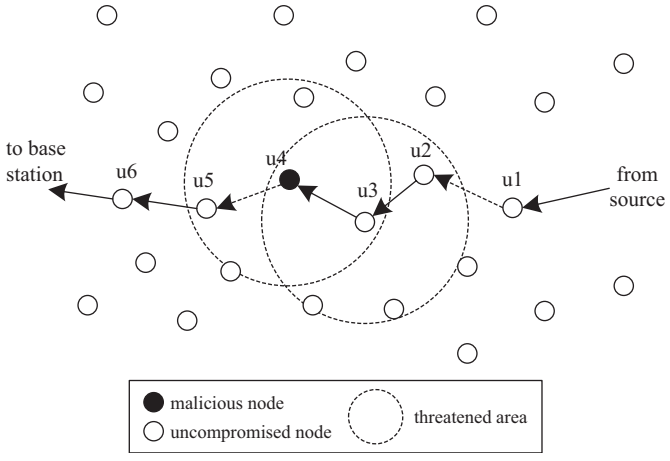


Fig. 6. Identification of suspect nodes.

(alert packets) provided by the intrusion detection scheme proposed in this paper.

Energy-efficient link-layer jamming attacks have been proved to be applicable in sensor networks [13,8]. If we suppose that malicious nodes have the potential to launch link-layer jamming attacks to indirectly cause packet loss between neighboring normal nodes, then we have to consider more about the threatened areas of suspect nodes. This is another area for our future work.

5. Simulation study

In this section, we evaluate the performance of our scheme through simulations that assume a more realistic scenario. In this scenario, we consider not only packet loss due to malicious dropping but also packet loss due to poor channel conditions. Our detection scheme is always working during each simulation run so that we can detect the attacks as soon as they happen.

This simulation scenario uses a field size of $2000 \times 2000\,\mathrm{m}^2$ where 400 nodes are uniformly distributed. One stationary sink and one stationary source sit on opposite sides of the field, with

about 20 hops in between. We carry out a simulation event in which the source generates 500 reports in total and one report is sent out every two seconds. Packets can be delivered hop-by-hop at 19.2 Kbps. To make our scheme more resilient in poor channel conditions, we implement a hop-by-hop transport layer retransmission mechanism beneath our scheme, which is quite similar to that in PSFQ [12]. The channel error rate is 10% by default, which is often regarded as a rather harsh channel condition. In order to avoid being detected, the malicious nodes drops only a small proportion of the packets passing by. To evaluate the performance of our scheme in the worst case, we suppose that to minimize the probability of being detected, the adversary compromise only consecutive nodes along a forwarding path, i.e., $q = 0$.

We propose the following metrics to evaluate the performance of our detection scheme.

- *Detection rate*: Namely, the detection probability, measures the ratio of the number of detected maliciously dropped packets to the total number of maliciously dropped packets including the undetected ones.
- *Mis-alert rate*: Namely, the false positive rate, measures the ratio of the number of detected lost packets, which are lost due to poor channel conditions, to the total number of detected lost packets.
- *Relative communication overhead*: Measures the ratio of the total communication overhead in a system that incorporates our detection scheme against a system that has nothing to do with the selective forwarding attacks (we call this *the base system*).

The detection rate and mis-alert rate mainly focus on the detection accuracy of our scheme, while the relative communication overhead tries to compare our detection scheme with other solutions defending against selective forwarding attacks.

5.1. Detection accuracy

In this subsection, we study how the detection accuracy of our scheme is affected by the channel error rate, malicious dropping rate, retransmission limit, k-covered acknowledgement, and the number of malicious nodes. The malicious dropping rate refers to the percentage of the total of packets maliciously dropped going through a malicious node.

Our first simulation shows the impact on the detection rate of k and the number of malicious nodes. Fig. 7(a) illustrates that the detection rate increases as k increases, but falls as the number of malicious nodes increase, so the detection rate of the simulation basically follows the theoretical expectation. Given $k = 2$, even when 25% of nodes (5 out of 20 intermediate nodes) are compromised, the detection rate is still about 90%. As we expected, as k increases, we also find a falling probability that enough malicious nodes are selected as checkpoints such that they can avoid detection. As a result, the detection rate increases. Please note that the theoretical results are supposed to be in perfect channel conditions but seems to be little different from the simulation results supposed to represent poor channel conditions. This suggests that channel error rate has little impact on detection rate, which will be further confirmed in our second simulation.

Our second simulation investigates the impact of the channel error rate and malicious dropping rate on the detection rate. Fig. 7(b) shows that both the channel error rate and the malicious dropping rate do not significantly affect the detection rate. After further investigation, we conclude that an increased channel error rate may cause more packets lost due to poor channel conditions, but it will not prevent the detection of malicious dropping.

Our third simulation tests the impact of the channel error rate and the malicious dropping rate on mis-alert rate, namely, the false positive rate. As shown in Fig. 7(c), both the channel error rate and the malicious dropping rate significantly affect the mis-alert rate. An increased channel error rate causes more packets lost yet it is difficult to distinguish maliciously dropped packets from packets that are lost due to poor channel conditions, so mis-alert rate inevitably increases. We believe that as long as a malicious node drops more than a normal node does at a certain channel error rate, the attacks are still detectable and false positive rate kept low. For example, given channel error rate equals to 15% and malicious dropping rate equals to 20%, the mis-alert rate is about 10% in Fig. 7(c) and the detection rate is over 95% in Fig. 7(b).

Finally, our fourth simulation indicates that transport layer retransmission mechanism can effectively lower the mis-alert rate by reducing the number of lost packets due to channel failure. As shown in Fig. 7(d), even when channel error rate is 15%, simulating a rather harsh channel condition, and malicious nodes drop only a very small proportion of packets (10%), if retransmission limit is set to 10, the mis-alert rate is still less than 10%.

5.2. Communication overhead

We use relative communication overhead to compare our scheme with other anti-selective-forwarding approaches such as multipath forwarding mentioned in [6]. The simulation results are based on the suggestion of the packet format in Fig. 2. We assume that the communication overhead of multipath forwarding is n times as much as the base system, where n is the number of paths of multipath forwarding. We regard the base system as a reference so that we can compare our approach with the multipath forwarding approach.

Fig. 8(a) investigates the impact of the channel error rate and malicious dropping rate on relative communication overhead. The three curves in Fig. 8(a) seem to closely overlap, which means that both channel error rate and malicious dropping rate do not affect relative communication overhead very much. Increased channel error rate does cause more packets lost and increase absolute communication overhead, but relative communication overhead is not affected.

Fig. 8(b) shows that k seems to the key factor that affects relative communication overhead. k decides how many ACK packets are transferred along the forwarding path. However, from the figure, we can see that neither $k = 2$ nor $k = 3$ will incur a significant relative communication overhead. Given

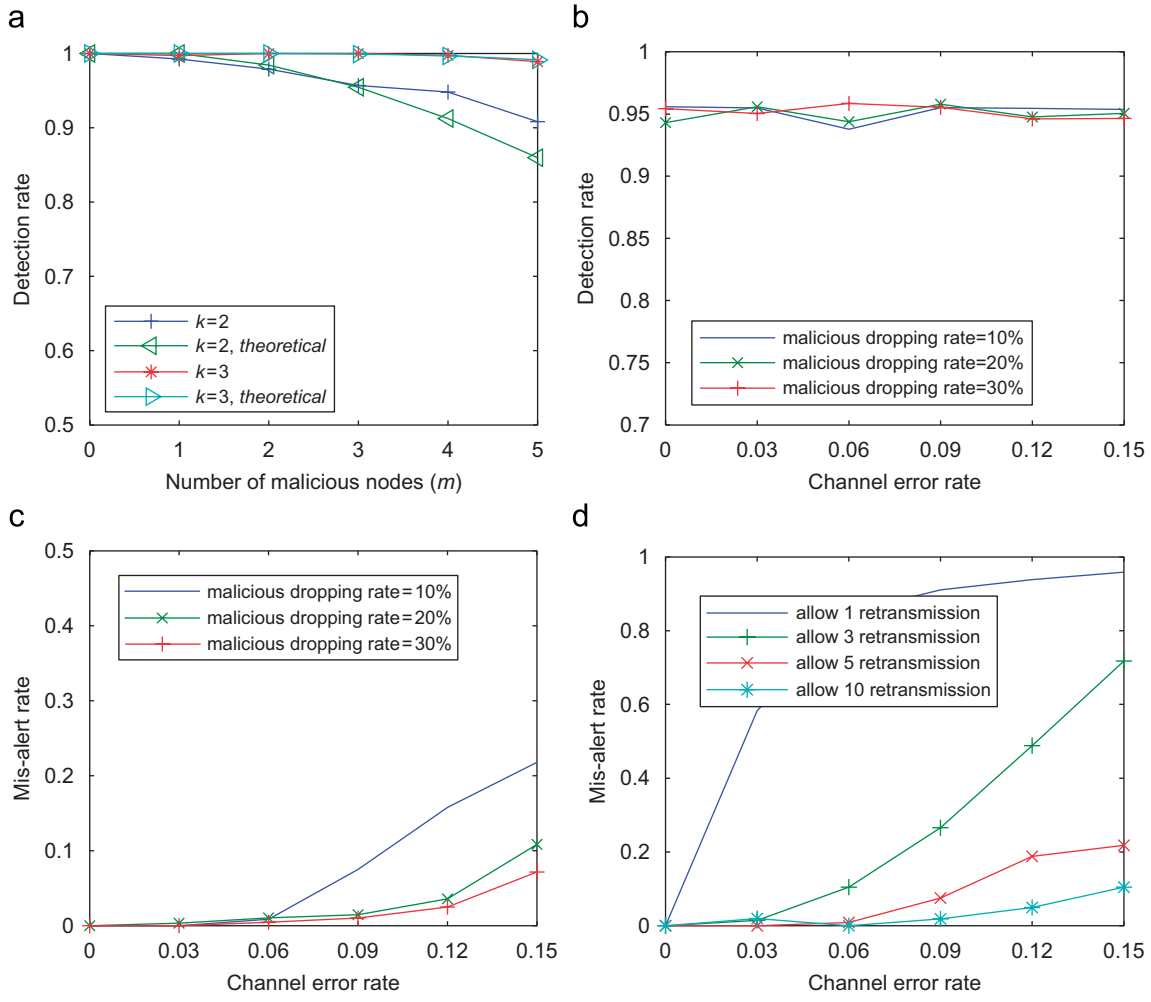


Fig. 7. Detection accuracy: (a) impact of m on detection rate, given *channel error rate* $= 10\%$; (b) impact of channel error rate on detection rate, given $k = 2$, $m = 3$; (c) impact of channel error rate on mis-alert rate, given $k = 2$, $m = 3$, *retransmission limit* $= 5$; (d) impact of retransmission limit on mis-alert rate, given $k = 2$, $m = 3$, *malicious dropping rate* $= 10\%$.

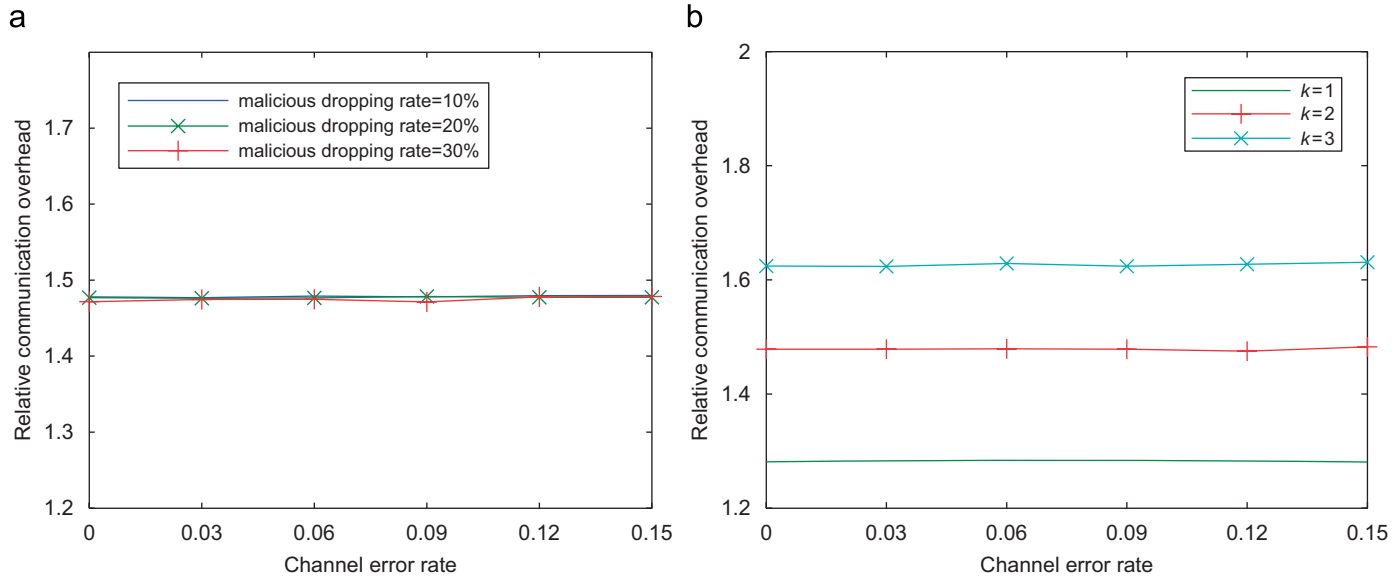


Fig. 8. Communication overhead: (a) impact of channel error rate and malicious dropping rate on communication overhead; (b) impact of k on communication overhead.

$k = 2$, the communication overhead of our detection scheme is less than 1.5 times of that of a system that does not incorporate our detection scheme.

It is really a tradeoff between communication overhead and detection capability. In our simulation settings, our communication overhead is less than 1.5 times of that in a base system. We think 50% more is acceptable for some security-sensitive applications. Obviously, in terms of communication overhead, our detection scheme shows its attractive advantage over the multipath forwarding approach, but this is not to suggest it supercedes multipath forwarding. Multipath forwarding is a prevention-based approach, while our scheme is a detection-based approach. Indeed, given the difficulty of preventing a compromised-yet-undetected node from dropping a single packet, a secure system should incorporate both approaches. For example, our detection scheme could be used to transfer ordinary packets, but when the source node generates a very important packet, it should be delivered to the base station through multiple paths.

6. Discussion

In this section, we first summarize several features of our scheme and then discuss some other potential improvements to assist in defending against selective forwarding attacks.

6.1. Features

Our detection scheme has two desirable features. First, checkpoint nodes, responsible for acknowledgement for safely received packets, are randomly selected. This random selection significantly increases the system resilience by preventing sensor nodes from becoming the targets of attempts to compromise them. Second, source nodes in our scheme are capable of collecting alert information. This source-side detection mechanism is so advantageous that even when the base station is deafened by surrounding malicious nodes, the source nodes remain capable of making decisions and responding.

However, several constraints still exists in our scheme. First, due to the random nature of our checkpoint selection mechanism, it is difficult to guarantee that exactly p percent of nodes are selected as checkpoint nodes. It is a dilemma that randomness guarantees the unpredictable selection, thus protecting sensor nodes from compromising, whereas, on the other hand, it is difficult to guarantee a fixed percentage for a given forwarding path. Second, if two compromised nodes, distributed far away from each other, can communicate with each other via a fast wormhole channel, it is still likely that they can cooperate to drop packets without detection.

In the next subsection, we propose several potential approaches to improve the detection performance of our scheme.

6.2. Potential improvements

6.2.1. Upstream detection and downstream detection

In our scheme, malicious dropping by malicious nodes downstream can potentially be detected by intermediate nodes upstream of malicious nodes. We call this upstream detection. We can also implement some downstream detection mechanisms, in which intermediate nodes downstream of malicious nodes are also responsible for detecting abnormal packet loss occurring upstream. For example, given that routes from the source nodes to the base station do not change frequently, an intermediate node is supposed to receive packets from upstream with consecutive packet IDs originating from a specific source node. Thus, if inconsecutive packet IDs arrive at the intermediate node, packet loss might have happened upstream and the intermediate node can generate an alert packet and have it delivered to the base station. Discontinuity of packet IDs might be caused by an upstream compromised node, a nearby outside jammer, or even by routing topology changes. Thus, in practice, it is likely that mis-alerts will be generated. However, as long as the base station ultimately receive the report packets, it is easy for the base station to remove false alerts. This has the added advantage of allowing the base station to collect alert information from intermediate nodes.

6.2.2. Incorporation of detection and prevention

Our scheme defends against selective forwarding attacks in a detection-based fashion, whereas multipath forwarding operates in a prevention-based fashion. Please note that we do not reject multipath forwarding approach, since redundancy is always an effective technique to increase system reliability. System performance can be further improved by incorporating intrusion detection and multipath forwarding. Sometimes, it can be quite difficult to prevent a compromised-yet-undetected intermediate node from dropping a single packet going through it. One answer might be to allow the source node to decide how a packet is delivered. If the packet is a very important one, the source node can have the packet delivered via multipath forwarding, or even flooding. If the packet is just a routine message, the packet can be delivered via one-path forwarding with intrusion detection.

6.2.3. Reason behind packet loss

Packet loss can be caused by compromised nodes, outsider jammers, as well as poor radio conditions, but without jamming detection techniques we find that it is difficult to discern the exact reason behind packet loss, as they exhibit similar symptoms. However, as long as the malicious nodes, including compromised nodes and outside jammers, cause more packet loss than a normal node does at a certain channel error rate, the attacks are detectable. In future work, we plan to integrate the jamming detection techniques (such as [13,8]) with our scheme so that we can determine the level of radio conditions and further find out the exact reason behind packet loss.

7. Related work

WSN security has in recent years been the subject of a number of proposals. Zhang and Lee [18] were among the first to study the problem of intrusion detection in wireless ad

hoc networks. Karlof et al. [6] analyze attacks against sensor network routing protocols and points out possible defenses. In the same paper, the author suggests a possible way to counter selective forwarding attacks by using multipath routing. Deng et al. [2] propose INSENS, an intrusion-tolerant scheme based on multipath routing. These schemes [6,2] are all based on redundant routing. The paper [16] did present a detection scheme for selective forwarding attacks. However, some problems still exist in [16]. For example, the members of a checkpoint list can be predicted, therefore making part of the intermediate nodes the target of compromising. In this paper, we make some improvements making the detection scheme more resilient against attacks by adopting the random-checkpoint-selection technique.

En-route filtering of injected false data in sensor networks has also been studied recently [19,15,3,14]. Zhu et al. [19] propose an interleaved key scheme, in which member nodes and intermediate nodes set up interleaved keys using randomly predistributed keys. The SEF scheme [15] proposed by Ye et al. tries to filter false data by using a probabilistic approach. Random keys are shared between the intermediate nodes and the source nodes in a sensor node group or cluster. Intermediate nodes can verify the MACs generated by the source nodes before forwarding packets. Yang et al. [14] present a more resilient approach based on location-binding keys. However, in his scheme, the relative position between source nodes and the base station is static. His scheme will be inefficient if there is more than one base station, or if the base station is mobile.

Other types of attacks in sensor networks have also attracted recent research attention. Secure time synchronization is studied in [4,9]. The feasibility of detecting jamming attacks is discussed in [8,1]. Detecting jamming attacks can help us to find out whether the original cause of packet loss is due to poor radio conditions, due to compromised nodes, or due to outside jammers.

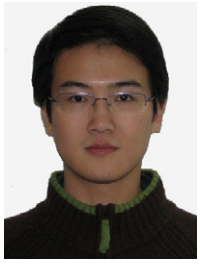
8. Conclusion

In this paper, we propose a simple and efficient security scheme for detecting selective forwarding attacks. Unlike common intrusion detection approaches in which detection is implemented in the base station or in a central controller, the source nodes in our scheme have the potential to detect selective forwarding attacks, which means that even when the base station is temporarily deafened by adversaries, attacks can still be detected by the source nodes. In order to reduce the communication overhead as well as to save the consuming energy in each sensor node, we can deliver packets normally in a leisure time period, only activating the detection scheme in some sensitive intervals. Several potential approaches remain to be taken to improve the defense capabilities of our scheme. For instance, use of downstream detection would help the base station to collect alert information; the incorporation of proper redundancy into our scheme would increase system resilience; and finally, the use of jamming detection techniques in combination with our proposed scheme would allow the identification of the original causes of packet loss.

References

[1] S. Capkun, J. Hubaux, Secure positioning of wireless devices with application to sensor networks, in: Proceedings of IEEE InfoCom, 2005, pp. 1917–1928.
[2] J. Deng, R. Han, S. Mishra, INSENS: intrusion-tolerant routing in wireless sensor networks, Technical Report, November 2002.
[3] J. Deng, R. Han, S. Mishra, Defending against path-based DoS attacks in wireless sensor networks, in: Proceedings of the Third ACM on the Security of Ad hoc and Sensor Networks (SASN 2005), 2005, pp. 89–96.
[4] S. Ganeriwal, S. Capkun, C. Han, M.B. Srivastava, Secure time synchronization service for sensor networks, in: Proceedings of the Fourth ACM Workshop on Wireless Security (WiSe 06), 2006, pp. 97–106.
[5] C. Intanagonwiwat, R. Govindan, D. Estrin, Directed diffusion: a scalable and robust communication paradigm for sensor networks, in: Proceedings of ACM MobiCom, 2000, pp. 56–67.
[6] C. Karlof, D. Wagner, Secure routing in wireless sensor networks: attacks and countermeasures, in: Proceedings of the 2003 IEEE International Workshop on Sensor Network Protocols and Applications, 2003, pp. 113–127.
[7] B. Karp, H. Kung, GPSR: greedy perimeter stateless routing for wireless networks, in: Proceedings of ACM MobiCom, 2000, pp. 243–254.
[8] Y.W. Law, L.V. Hoesel, J. Doumen, P. Hartel, P. Havinga, Energy-efficient link-layer jamming attacks against wireless sensor network mac protocols, in: Proceedings of the Third ACM on the Security of Ad hoc and Sensor Networks (SASN 05), 2005, pp. 76–88.
[9] M. Manzo, T. Roosta, S. Sastry, Time synchronization attacks in sensor networks, in: Proceedings of the Third ACM Workshop on Security of Ad hoc and Sensor Networks, 2005, pp. 107–116.
[10] A. Perrig, R. Szewczyk, V. Wen, D. Culler, D. Tygar, SPINS: security protocols for sensor networks, in: Proceedings of ACM Mobicom, 2001, pp. 189–199.
[11] Y. Sankarasubramaniam, O.B. Akan, I.F. Akyildiz, ESRT: event to sink reliable transport in wireless sensor networks, in: Proceedings of ACM MobiHoc, 2003, pp. 177–188.
[12] C.Y. Wan, A.T. Campbell, L. Krishnamurthy, PSFQ: a reliable transport protocol for wireless sensor networks, in: Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications, 2002, pp. 1–11.
[13] W. Xu, W. Trappe, Y. Zhang, T. Wood, The feasibility of launching and detecting jamming attacks in wireless networks, in: Proceedings of ACM MobiHoc, 2005, pp. 46–57.
[14] H. Yang, F. Ye, Y. Yuan, S. Lu, W. Arbaugh, Toward resilient security in wireless sensor networks, in: Proceedings of ACM MobiHoc, 2005, pp. 34–45.
[15] F. Ye, H. Luo, S. Lu, L. Zhang, Statistical en-route filtering of injected false data in sensor networks, in: Proceedings of IEEE InfoCom, 2004, pp. 2446–2457.
[16] B. Yu, B. Xiao, Detecting selective forwarding attacks in wireless sensor networks, in: Proceedings of the Second International Workshop on Security in Systems and Networks (IPDPS 2006 Workshop), 2006, pp. 1–8.
[17] Y. Yu, D. Estrin, R. Govindan, Geographical and energy-aware routing: a recursive data dissemination protocol for wireless sensor networks, UCLA Computer Science Department Technical Report, UCLA-CSD TR-01-0023, Technical Report, 2001.
[18] Y. Zhang, W. Lee, Intrusion detection in wireless ad-hoc networks, in: Proceedings of ACM MobiCom, 2000, pp. 275–283.
[19] S. Zhu, S. Setia, S. Jajodia, N. Peng, 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, 2004, pp. 259–271.

Bin Xiao received the B.Sc. and M.Sc. degrees in Electronics Engineering from Fudan University, China in 1997 and 2000, respectively, and Ph.D. degree from University of Texas at Dallas, USA, in 2003 from Computer Science. Now he is an Assistant Professor in the Department of Computing of Hong Kong Polytechnic University, Hong Kong. His research interests include communication and security in computer networks, peer-to-peer networks, wireless mobile ad hoc and sensor networks.



Bo Yu received his Ph.D. degree in Computer Science from Fudan University, China, in 2006. During 2005-11-2006–8, he worked in Hong Kong Polytechnic University as a research assistant. Currently, he is working in Wayne State University as a postdoc fellow. His research interests are in computer networks, especially in Mobile Ad hoc Networks, including wireless sensor networks, vehicular networks, etc.

Chuanshan Gao graduated from Fudan University, China, in 1963. Since then, he has been working in the same university. During 1981–1983, he was a visiting scholar in Department of CS at UIUC, USA. Now he is director of C & C Lab and professor of Computer Science & Engineering Department at Fudan University. His research interests focus on Data Communication, Computer Networks, Distributed System and their applications. He led and participated in many research projects on the topics mentioned above which have been awarded several times by Province Governments or State Ministries of China. More than 150 papers have been published, most in Chinese and some in English in well-known international conferences and journals.