Replication Attack Detection with Monitor Nodes in Clustered Wireless Sensor Networks

Guo Cheng*, Songtao Guo*, Yuanyuan Yang‡ and Fei Wang*

*College of Electronic and Information Engineering, Southwest University, Chongqing, 400715 China
‡Department of Electrical & Computer Engineering, Stony Brook University, Stony Brook, NY 11794, USA

***Abstract*—Wireless sensor networks (WSNs) are often deployed in hostile environments where an adversary may physically capture some of the nodes in WSNs, and replicate them in a large number of clones, easily taking control of networks. A few solutions have been proposed to cope with this problem. However, these solutions cannot adapt to the change of the network size and have low detection efficiency for clone nodes. In order to discover the clone nodes fast, in this paper, we propose an improved LEACH (NI-LEACH) protocol to reduce the scale of the cluster by considering the residual energy of nodes and the optimal number of clusters. Furthermore, we design an intrusion detection algorithm to detect the replication attacks by introducing monitor nodes in the network so as to greatly reduce the occurrence of tampering with the information. Simulation results show that our proposed algorithm is simple yet efficient. An attacker can be detected with high probability while achieving approximately optimal throughput. The network's ability against the attack from clone nodes is greatly improved.**

***Index Terms*—Wireless sensor networks, node attacks, replication, intrusion detection algorithms, monitor nodes.**

I. Introduction

Wireless sensor networks (WSNs) are composed of a group of sensor nodes with limited resources [1]–[3]. WSNs are usually deployed in harsh environments to fulfill military or civil tasks [4]. Due to their operating nature, they are often unattended and generally lack effective ways against the tamper attack, hence they are vulnerable to most of new types of attacks. For example, an adversary could capture some network nodes, called *clone nodes*, to acquire the information stored there and replicate the messages transmitted by them, even tamper the local message such that it is difficult to find those clone nodes. Thus it is critical to ensure the security of wireless sensor networks.

In practice, sensor nodes can be easily captured [4], [5], because they are usually unprotected by physical shielding due to cost considerations [6], and are often unattended after deployment. If we cannot detect these replicas, the network will be vulnerable to a large number of internal attacks [7]. The threat of clone attack can be characterized from two aspects. First, a clone node is usually considered to be honest by its neighbors. In fact, without global countermeasures, honest nodes cannot be aware of the fact that there is a clone node among their neighbors. Second, besides the information of clone nodes can be copied, it can also be tampered with. Once a node has been captured and compromised, the attack will be sustained. It is very easy to make further clones of the same node. Fig. 1 illustrates the node replication attack in a WSN. After the original node is captured by the attacker, all information is taken from the original node. The attacker then re-inserts this captured node to the network without any change [8].

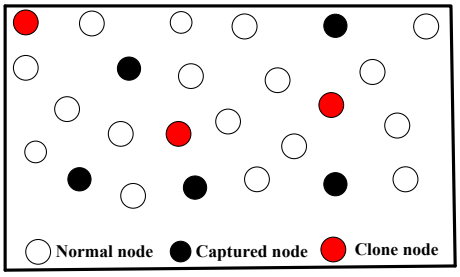

Fig. 1. A WSN with clone nodes.

There has been some work in the literature [9], [10] on node attack detection methods and detection of node replication in static WSNs. However, most of the existing clone detection methods cannot adapt to the change of the network size and have low detection efficiency for clone nodes. Moreover, although most of the methods are easy to implement in a centralized manner, they cannot handle the attack in which both the data transmission nodes and the cluster head nodes are captured at the same time. For large-scale WSNs, it is difficult to find the positions of clone nodes since they may be at any position in the network. In order to efficiently find the clone nodes, we need to reduce the scale of the cluster by appropriate clustering. However, most existing clustering protocols including LEACH select cluster heads in a random manner and do not consider the optimal number of clusters in large-scale WSNs.

In this paper, we first propose an improved LEACH (NI-LEACH) protocol to determine the optimal scale of the cluster and enhance the detection efficiency. Compared to the original LEACH protocol, our proposed NI-LEACH protocol has the following features. First, we consider the optimal number of clusters in a network, which not only affects the energy consumption of data transmission, but also determines the efficiency of discovering clone nodes. Furthermore, to ensure energy balance, we introduce the residual energy of nodes in the NI-LEACH so that in each round a node with more energy should have higher probability to become a cluster head.

978-1-4673-8590-9/15/$31.00 ©2015 IEEE

Furthermore, we design an intrusion detection algorithm to address the problem of replication attacks, by quickly determining the replicated nodes in the clustered network. The intrusion detection algorithm consists of four steps: preprocessing, selecting monitor nodes, observing data transmission nodes, and monitoring cluster head nodes. In order to improve the accuracy of detection, we also introduce the concept of monitor nodes in our algorithm so that we can observe the message transmission and the behavior of cluster heads. Simulation results show that our algorithm is effective to detect the replication attack of nodes.

The rest of the paper is organized as follows. In Section II, we introduce the related work. Then in Section III, we propose an improved LEACH protocol. In Section IV, we propose an intrusion detection algorithm. Section V provides simulation results and the corresponding discussions. Finally, we conclude this paper in Section VI.

II. Related Work

In this section, we review the previous methods of detecting clone nodes in WSNs.

One of the solutions for the detection of clone attacks is based on centralized Base Station (BS) [11]. In this solution, each node sends a list of its neighbors and their locations (that is, the geographical coordinates of each node) to a BS. Clone detection is to find the same node ID in two lists with inconsistent locations. Then, the BS revokes the clones. however, this solution has several drawbacks, such as the presence of a single point of failure (the BS) and high communication cost due to the large number of messages. Other solutions rely on local detection. For example, in [11], a voting mechanism is used within a neighborhood to agree on the legitimacy of the node. However, this method fails to detect clones that are not within the same neighborhood.

In [12], messages are collected in a promiscuous mode, and pre-selected rules are applied to determine if a failure occurs. An intrusion alarm is raised if the number of failures exceeds a predefined threshold. Choi et al. [13] proposed a detecting node clones method called *SET* to detect the abnormality that an ID appears in different exclusive subgroups. However, SET may have false detections when insidious leaders in the trees forge IDs not in their clusters. Xing et al. [14] proposed a method to detect the abnormality that a node has different fingerprints. Fingerprint is generated from node's neighbor list, and the BS detects the replicas if it receives different fingerprints for the same ID. However, this scheme requires each node to periodically communicate with the BS. In [15], the authors proposed an end-to-end detection of wormhole attack (EDWA) in wireless ad-hoc networks. They first presented the wormhole detection which is based on the smallest hop count estimation between the source and the destination. However, the drawback of end-to-end detection is that if the data transmission is tampered in the midway, it is difficult to find the clone nodes. A technique to overcome this shortcoming is the monitor mechanism proposed in [16], which is on the basis of many misbehavior detection algorithms and trust or reputation systems. Another monitor mechanism was proposed in [17], where next-hop node's behavior is measured by the local evaluation record. The trust level of a node is characterized by the combination of its local observation and the broadcast information. Although many ad hoc trust or reputation systems [18]–[21] adopt different trust level calculation mechanisms, their basic process is similar to that in [17], including monitoring, broadcasting local observation, combining the direct and indirect information into the final trust level.

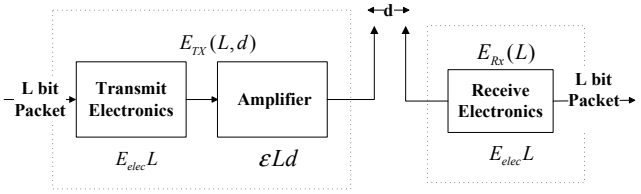

Fig. 2. Radio energy dissipation model.

III. NI-LEACH

In this section, we propose an improved LEACH protocol called *NI-LEACH*.

A. Optimum number of clusters

Previous works [22], [23] have studied that the optimum probability that a node is elected as a cluster head is a function of spatial density when nodes are uniformly distributed over the sensing field. We use the energy model similar to that in [22]. According to the radio energy dissipation model shown in Fig. 2, in order to achieve an acceptable signal-to-noise ratio (SNR), when transmitting an L bit message over distance d, the energy consumed by the radio is given by

$$E_{Tx}(l,d) = \begin{cases} L * E_{elec} + L* \in_{fs} *d^2 & if\ d \le d_0 \\ L * E_{elec} + L* \in_{mp} *d^4 & if\ d > d_0 \end{cases} \quad (1)$$

and when receiving this message, the energy consumption can be expressed by

$$E_{Rx}(l) = L \times E_{elec} \quad (2)$$

where E_{elec} is the energy dissipated per bit on the transmitter or the receiver circuit, $\in_{fs}$ and $\in_{mp}$ denote the energy loss rate per unit of data transmission of the transmitter for $d \le d_0$ and $d > d_0$, respectively, which depend on the transmitter amplifier model, d is the distance between the sender and the receiver, and d_0 denotes a constant distance threshold. Assume that n nodes are uniformly distributed in an area of $A = M \times M$ square meters and there are k clusters in this area. Thus each cluster has on average n/k nodes (one cluster head and $n/k-1$ cluster member nodes).

In the following, we first consider the case that the distance of member nodes to the cluster head/the sink is less than or equal to d_0, i.e., $d \le d_0$. In this case, the energy dissipated in the cluster head node during a round of selecting CH node is calculated by

$$\begin{aligned} E_{CH} = (n/k-1)L * E_{elec} + n/k * L * E_{DA} \\ +L * E_{elec} + L* \in_{fs} *d_{toBS}^2 \end{aligned} \quad (3)$$

where E_{DA} is the processing (data aggregation) cost of a bit transmitted to the sink, and d_{toBS} is the average distance between the cluster head and the sink. Thus, the energy used in each member node is computed by

$$E_{nonCH} = L * E_{elec} + L* \in_{fs} *d_{toCH}^2 \tag{4}$$

where d_{toCH} is the average distance between a cluster member and its cluster head. Since the nodes are uniformly distributed, d_{toCH} can be given by

$$\begin{aligned} d_{toCH}^2 &= \int_0^{x=x\max}\int_0^{y=y\max}(x^2+y^2)\varphi(x,y)d_x d_y \\ &= \frac{M^2}{2\pi k} \end{aligned} \tag{5}$$

where $\varphi(x,y)$ is the node distribution density function. Thus the energy dissipated in a cluster per round is calculated by

$$E_{cluster} = E_{CH}+(n/k-1)E_{nonCH} \approx E_{CH}+n/k*E_{nonCH} \tag{6}$$

The total energy dissipated in the network is

$$\begin{aligned} E_{total} &= k * E_{cluster} \\ &= L(2nE_{elec} + nE_{DA}+ \in_{fs} (kd_{toBS}^2 + nd_{toCH}^2)) \end{aligned} \tag{7}$$

By setting the derivative of E_{total} with respect to k to zero, the optimal number of clusters is computed by

$$k_{opt} = \sqrt{\frac{n}{2\pi}}\frac{M}{d_{toBS}} = \sqrt{\frac{n}{2\pi}}\frac{2}{0.765} \tag{8}$$

The equality on the right-hand side is based on the result that the average distance from a cluster head to the sink is given by [23]

$$d_{toBS} = 0.765\frac{M}{2} \tag{9}$$

It is interesting to find that for a given sensing area, the optimal number of clusters is independent of the dimensions of the field and only depends on the number of nodes n.

Now, we consider the total energy dissipated in the network in the case that the distance of any node to the sink or its cluster head d is more than d_0, i.e., $d > d_0$. Similar to the derivation for (8), we can obtain the optimal number of clusters in the case of $d > d_0$

$$k_{opt} = \sqrt{\frac{n}{2\pi}}\sqrt{\frac{\in_{fs}}{\in_{mp}}}\frac{M}{d_{toBS}^2} \tag{10}$$

The optimal probability of a node being selected as a cluster head, p_{opt}, can be given by

$$p_{opt} = \frac{k_{opt}}{n} \tag{11}$$

The optimal number of clusters is very important since it not only affects the energy consumption of data transmission, but also determines the efficiency of discovering clone nodes. Therefore, one of our contributions is to give the closed-form expression of the optimal number of clusters in a sensing field. We can observe from (8) and (10) that the optimal number of clusters depends on the number of nodes, the size of sensing field and the average distance of nodes to the sink. Clearly, the farther average distance will result in less clusters.

B. Residual energy of nodes

In order to balance network load, nodes with more residual energy should be much more active. To ensure energy balance, the nodes with more residual energy should have higher probability to be elected as cluster heads so that the residual energy of nodes will be more uniform. Now, we introduce the residual energy of node i as

$$E_p(i) = \begin{cases} 1 - \frac{E_{netorg}}{E_r(i)} & \forall E_{netorg} < E_r(i) \\ 0 & otherwise \end{cases} \tag{12}$$

where E_{netorg} is the average residual energy of all the nodes, and $E_r(i)$ denotes the residual energy of node i.

C. NI-LEACH

In the following, we propose NI-LEACH by introducing the optimal probability of a node being a cluster head, p_{opt}, in (11) and the residual energy of node, E_p, in (12). Initially each node becomes a cluster head with probability p_{opt}. On average, thus, there are $n\times\ p_{opt}$ nodes to become cluster heads per round for an epoch.

Let G denote the set of non-elected nodes. In order to maintain a steady number of cluster heads for each round of rotating, the probability of nodes in set G becoming a cluster head increases after each round in the same epoch. At the beginning of each round, each node in set G independently makes a decision of choosing a random number in $[0, 1]$. If the chosen random number is less than a threshold $T(s)$, then the node becomes the cluster head in the current round. The nodes with more residual energy in set G have higher probability to become cluster heads by increasing the threshold $T(s)$. To balance the energy load, we consider the residual energy in computing threshold $T(s)$. The threshold is calculated by

$$T(s) = \begin{cases} \frac{p_{opt}}{1-p_{opt}(r \bmod \frac{1}{p_{opt}})} + E_m * E_p(s) & if\ s \in G \\ 0 & otherwise \end{cases} \tag{13}$$

where r is the current round number (starting from round 0), and E_m is the weight factor, $E_m \in [0, 1]$. It is not difficult to observe from (13) that the probability of nodes in G becoming cluster heads increases with the number of rounds in the same epoch and equals 1 in the last round of the epoch. It is worth noting that compared to LEACH [24], where the optimal probability p_{opt} needs to be determined a priori and the energy of sensor nodes is not considered, NI-LEACH selects cluster heads by both the p_{opt} and the residual energy $E_p(s)$.

IV. Intrusion Detection Algorithm

In a WSN, an attacker may change the network topology at any time. It can replicate or tamper the information so that the clone nodes have the same permissions as the valid nodes to communicate with all the nodes. In the following, we propose an intrusion detection algorithm to detect and revoke the compromised nodes to improve detection probability. This algorithm consists of four steps, namely,

- Pre-processing;
- Selecting the monitor nodes;

- Observing the data transmission of nodes;
- Monitoring the cluster head nodes;

Next, we will describe the four steps of the intrusion detection algorithm in detail.

A. Pre-processing

This step aims to cluster the considered WSN by the proposed NI-LEACH protocol in Section III. We assume that the sensor network consists of n sensors with IDs, $(1, 2, \ldots, n)$, and all the nodes are distributed uniformly in the sensing area. In addition, the initial energy of nodes are assumed to be equal. In this process, equation (13) is used for the rotation of cluster heads. It is known that the node energy consumption rate is different due to the different responsibility of each node. Thus we define a cost function $W(S_i)$ to represent the energy consumed by sensor s_i

$$W(s_i) = 1 - \frac{e_i(t)}{e_i(0)} \tag{14}$$

where $e_i(t)$ is the residual energy of sensor s_i at time t and $e_i(0)$ is the initial energy of the sensor.

B. Selecting the monitor nodes

Once the clusters are determined in the considered network, the next step of the intrusion detection algorithm is to select the monitor nodes. In the previous work, in order to efficiently mitigate the misbehavior of the malicious nodes in a WSN, several misbehavior detection algorithms were proposed for the trust or reputation system, where each cluster has only one monitor node to find clone nodes. However, in practice, multiple monitor nodes can reduce the miss-detection probability and the energy consumption of all the nodes. This is because that different monitor nodes are responsible for observing the transmission and behavior of different nodes or cluster heads (CHs). Therefore, it is important to determine the appropriate number of monitor nodes. In our proposed algorithm, the monitor nodes are determined by taking into account of both energy consumption and detection range of nodes. All monitor nodes in a cluster should cover the cluster. Our objective is to find an appropriate set of monitor nodes, denoted by S, in each cluster so as to minimize the total energy consumption of m nodes, which is define by $Cost(H) = \sum_{i=1}^{m} W(s_i)$. m denotes the number of the elements in set S. The procedure of finding set S is given by the following algorithm 1.

C. Monitoring the data transmission of nodes

Once the set of monitor nodes (S) is determined, each node within the coverage of a monitor node will be monitored in data transmission. For easy understanding, here we consider a simple model to illustrate how to find the abnormal nodes and reduce the miss-detection probability. In a data transmission flow network shown in Fig. 3, there are four types of nodes: the source node T, the cluster node C, the attacker R, and the monitor nodes M_1 and M_2. The solid line denotes transmission from T to C, where the transmission is relayed by the attacker R. The dashed lines denote the overhearing from monitor nodes to other nodes.

Algorithm 1 Selecting the monitor nodes

Input:
$B = (s_1, s_2, s_3, \ldots, s_N)$ is the set of sensors distributed randomly in cluster H;
v: the set of m nodes;
V: the set of sets v;
s': the set of candidate monitor nodes that can cover cluster H.
S': the set of sets s';

Output: Set of monitor nodes S such that $Cost(H)$ is minimized.

1: **for** each set $v \subset V$ **do**
2: **if** v cannot cover the region of H **then**
3: $S' \leftarrow null$;
4: **else**
5: $S' \leftarrow v$;
6: **end if**
7: calculate the $Cost(H) = \sum_{i=1}^{m} W(s_i)$ of each set $s' \in S'$;
8: select the minimum $Cost(H)$ and the corresponding set s';
9: $S \leftarrow s'$
10: **end for**
11: **return** the set S.

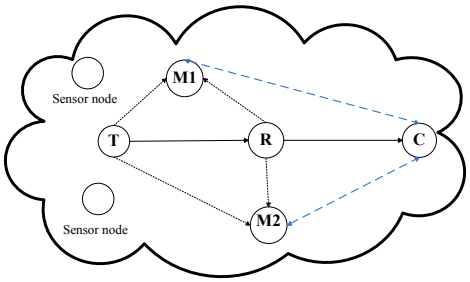

Fig. 3. Data transmission flow network.

We assume that the data transmission rate of each link is 1 packet per unit time, and the monitor nodes M_1 and M_2 know whether the information from T to C is tampered by the attacker R. We assume that data transmission from T to C is reliable. And, the monitor node supervises all data transmissions in the network with observation probability p. The source node T adopts maximum distance separable (MDS) code, where the data packets with the length y are encapsulated into the coded packets with the length x and $y < x$ using a (x, y) encoder function. With a (x, y) MDS code, we can know that the minimum hamming weight (d) of the (x, y) encoder function [25] is $d \leq x - y + 1$ through the lemma of singleton bound [26]. Therefore, the attacker R will always be found as long as more than $x - y$ bits of a packet are altered; otherwise, the attacker R will avoid being detected. Hence, the more bits of a packet are tampered, the more likely the attacker will be detected by monitor nodes. For

simplicity, we assume that the attacker will not be detected, that is, there will always no more than $x - y + 1$ bits of a packet be detected. Thus, it is easily to know the probability that an attacker cannot be detected is:

$$P_{miss}(x, y, p) = (1 - p)^{x-y+1}, \tag{15}$$

where $P_{miss}(x, y, p)$ denotes the miss-detection probability that the attacker R cannot be found by one monitor node. In this case, we construct a (x, y) encoder function such that

$$y = x + 1 - \frac{f(x, p)}{p}. \tag{16}$$

Thus, by Eq. (15), $P_{miss}(x, y, p)$ can be written as

$$P_{miss}(x, y, p) \leq e^{-p(x-y+1)} = e^{-f(x,p)}. \tag{17}$$

Furthermore, the probability that the attacker R cannot be found by g monitor nodes simultaneously, denoted by $P_{miss}(x, y, p, g)$, is

$$P_{miss}(x, y, p, g) = e^{-g*f(x,p)}. \tag{18}$$

To make $P_{miss}(x, y, p, g)$ arbitrarily small and y/x approach arbitrarily to optimum, a function $f(x, p)$ conforming to the actual situation needs to be appropriately chosen. In this paper, we construct $f(x, p) = \beta \ln x$ for any positive constant β. Then, we can get

$$\begin{aligned} P_{miss}(x, y, p, g) &\leq e^{-g*\beta \ln x} \\ &= x^{-g\beta} \to 0 \text{ as } x \to \infty, \end{aligned} \tag{19}$$

which is because $g > 0$, $x \to \infty$, and $x^{-g\beta} \to 0$, $P_{miss}(x, y, p, g) \to 0$. Using the above error detection code $P_{miss}(x, y, p, g)$, the coding rate y/x can be computed by

$$\begin{aligned} \frac{y}{x} &= \frac{x+1-\frac{\beta \ln x}{p}}{x} \\ &= 1 + \frac{1}{x} - \frac{\beta \ln x}{px} \to 1 \text{ as } x \to \infty. \end{aligned} \tag{20}$$

Thus, by integrating monitor nodes into a cluster, the incentive of an attacker to tamper information can be reduced by finding an appropriate β and making x sufficiently large.

Now, we analyze the benefit of deploying multiple monitor nodes by exploring the trade-off between throughput and security. Consider the network as shown in Fig. 3, where monitor nodes M_1 and M_2 can overhear data transmission of all links. For Fig. 3, we assume a slotted aloha access protocol with access probability α is used. To simplify the analysis, we further assume that a node will access the channel by transmitting dummy packets when it has no data packets to send. Under these assumptions, the throughput of each flow can be computed by

$$T = \alpha(1 - \alpha). \tag{21}$$

The probability that the transmission from T to R is successful and M can overhear the transmission is $(1 - \alpha)$. Meanwhile, the probability that M overhears the transmission from R to C while T remains silent is also $(1 - \alpha)$. Therefore, the probability with which the monitor node can successfully detect whether a packet has been tampered is given by

$$p = (1 - \alpha)^2. \tag{22}$$

The reason that the exponent in Eq. (22) is 2 is that we use a slotted access protocol. Similar to the data transmission flow in Fig. 3, P_{miss} can achieve arbitrarily small by choosing

$$y = x + 1 - \frac{\beta \ln x}{(1 - \alpha)^2}. \tag{23}$$

Then, the effective throughput is

$$T_E = T \times \frac{y}{x} = \alpha(1 - \alpha)(1 + \frac{1}{x}) - \frac{\alpha\beta \ln x}{(1 - \alpha)x}. \tag{24}$$

We can observe from (23) and (24) that the proposed scheme achieves a high level of security while maintaining a reasonably good throughput. Algorithm 2 summarizes the process of monitoring the data transmission of nodes.

Algorithm 2 Monitoring the cluster head nodes

Input:

source node T, cluster node C, attacker R, and set of monitor nodes (S);

Output:

miss-detection probability $P_{miss}(x, y, p, g)$, coding rate y/x and effective throughput T_E;

1: **if** attacker R cannot be observed by a monitor node in (S) **then**
2: $P_{miss}(x, y, p) \leftarrow (1 - p)^{x-y+1}$;
3: **else**
4: $P_{miss}(x, y, p) \leftarrow 1$;
5: **end if**
6: calculate probability $P_{miss}(x, y, p, g)$ by (18);
7: calculate coding rate y/x by (20);
8: calculate effective throughput T_E by (24);
9: **return** $P_{miss}(x, y, p, g)$, y/x, T_E.

D. Monitoring the cluster head nodes

It is complex to presume whether a CH is an intruder only according to the detection alarm message from one monitor node. Therefore, the detection alarm messages may be trusted if there are more than one monitor nodes detecting the misbehavior of one CH. Based on these considerations, we utilize m monitor nodes cooperatively monitoring one CH in a cluster. In addition, compared to the previous works, where revoking messages are flooded into the entire WSN, the revoking messages in our proposed detection approach are only flooded in the local cluster. After an abnormal CH is revoked by monitor nodes, it cannot communicate with other sensor nodes anymore. The revoking procedure of an abnormal CH can be described as follows.

TABLE I
ALARM TABLE IN EACH SENSOR NODE

Abnormal node	Alarm count	Current monitor node
U	1	S_i

1) Issuing alarm message: The core of this process is that a CH (u) broadcasts the message of its ID and location (l) encrypted with the cluster key K_C by Eq. (25), denoted by $\{Msg\}_{K_C}$, to each monitor nodes (MN) S_i in set S. After the MNs receive the message including ID and l, MNs in a cluster take turns to monitor CH node u for a period of time t. When a monitor node $S_i \in S$ detects the misbehavior of CH (u), it issues an alarm message $Alarm\{u\}$ shown in Eq. (26) to inform other monitor nodes in set S. If a monitor node receives more than X alarm messages, then it revokes the abnormal CH.

$$u \rightarrow S_i : \{Msg\}_{K_C} = \{ID, l\}_{K_C} \tag{25}$$

$$S_i \rightarrow S : \{Alarm\{u\}, S_i, X\} \tag{26}$$

2) Updating alarm table: Each monitor node maintains an alarm table shown in Table I to record the received alarm messages. An alarm table consists of three fields. The first field, "Abnormal node" records the ID of the suspected CH. The second one, "Alarm count" counts the number of alarm messages issued by different monitor nodes for the same suspected CH. The third field, "Current monitor node" lists the monitor node S_i of the received alarm messages. The alarm message in Eq. (26) is recorded as the second row in Table I.

3) Determination of alarm threshold X: Once the monitor nodes are attacked in clustered WSNs, the probability that a monitor node is compromised, P_c, depends on the actual deployment environment. Assume $(P_c)^X$ represents the probability that X monitor nodes are compromised simultaneously. The greater X is, the smaller $(P_c)^X$ will be. This is because P_c is less than one. Alarm threshold, X, should be appropriately determined, because if X is too large, it is hard to detect the misbehavior of one abnormal CH; in contrast, the detection alarm is too sensitive. Therefore, the alarm threshold, X, is determined by the tolerance factor of security θ and the probability of a monitor node being compromised P_c, which is calculated by

$$(P_c)^X < \theta. \tag{27}$$

4) Determination of the number of monitor nodes: The number of monitor nodes m plays an important role in network security and energy consumption. If there are only a small number of monitor nodes to work in the network, the monitoring time of each monitor node increases remarkably up to the limit t of a cycle. Moreover, each monitor node will consume more energy to monitor a CH, and could reduce the network security. Therefore, an appropriate m needs to be determined by considering the tradeoff between energy and security. Let P_f denote the probability of that a monitor node fails to detect the misbehavior of an attacker. And P_D denotes the probability that a monitor node successfully detects an attacker. Clearly, P_D is the function of P_f, X, and m, which is given by

$$P_D = \sum_{i=X}^{m} \binom{m}{i} (1 - P_f)^i P_f^{m-i}. \tag{28}$$

The process of monitoring the cluster head nodes can be summarized in Algorithm 3.

Algorithm 3 Monitoring the cluster head nodes

Input: CH(u) and the set of monitor nodes (S)
Output: Z: the set of abnormal cluster heads

A CH (u) broadcasts the message of its ID and location (l) encrypted with the cluster key K_C to each monitor nodes (MN) S_i, namely, $u \rightarrow S_i : \{Msg\}_{K_C} = \{ID, l\}_{K_C}$
for each monitor node $S_i \in S$ **do**
  if node S_i finds the misbehavior of CH (u) **then**
    $S_i \rightarrow S : \{Alarm\{u\}, S_i, X\}$;
    increase the "Alarm count" by 1 in alarm table of other monitor nodes;
  else
    node S_i does not send alarm message $Alarm\{u\}$ to other monitor nodes in set S;
    the message of each monitor node in the alarm table remain unchanged;
  end if
  if in the alarm table of any monitor nodes, alarm count exceeds the alarm threshold, X **then**
    $Z \leftarrow$the cluster head(u);
  else
    $Z \leftarrow null$;
  end if
end for
return the set Z

V. SIMULATION RESULTS

In this section, we evaluate our intrusion detection algorithm in terms of monitoring the data transmission of nodes and monitoring the cluster head nodes. Meanwhile, we compare our intrusion detection algorithm with the algorithm with one monitor node and without clustering. We consider a wireless sensor network in which 100 nodes are uniformly distributed in a $300m \times 300$ square area. The total energy of each node is $12J$.

A. Performance in monitoring the data transmission of nodes

Fig. 4 shows the miss-detection probability versus the observation probability p for a given packet length $x = 80$ and different β. It can be seen from Fig. 4 that the miss-detection probability decreases as the observation probability p increases, which means that attacker nodes will have higher probability to be caught. Moreover, we also find that the larger β are, the smaller the miss-detection probability is. Therefore, we can constantly enhance the observation probability p and β to achieve optimal efficiency in practical environments.

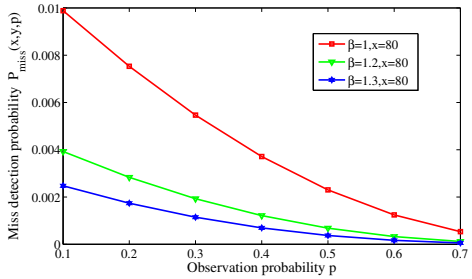

Fig. 4. Miss-detection probability versus observation probability p for different β.

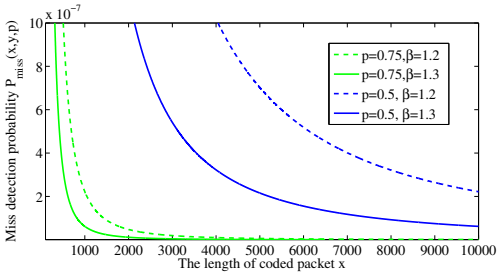

Fig. 5. Miss-detection probability versus the length of coded packet x with $y = x + 1 - \frac{\beta \ln x}{p}$.

Fig. 5 depicts the probability of miss-detection over the length of a packet x. A large x means the high coding/decoding complexity. We can observe that the miss-detection probability decreases as x increases, which is because the more coded packets are, the longer time the attack spend in corrupting the packets. As a result, the attacker have to tamper more messages in order to corrupt the packets, which makes it easier to be detected by the monitor nodes. In addition, we can observe from Fig. 6 that the miss-detection probability decreases quickly with the increase of the number of monitor nodes g. Therefore, in order to prevent nodes from being cloned, this feature can be used to determine the way of node deployment.

Fig. 7 illustrates the effective throughput versus channel access probability α for different x and β. Clearly, there exists a maximum effective throughput for all the cases. For example, in the case of $x = 60$ and $\beta = 2$, the effective throughput can be maximized and on average taken 0.14 packets per slot when α is about 0.3. Although the throughput is higher without source coding, it comes at the cost of not being able to provide any security guarantee. On the contrary, our scheme guarantees an upper bound for P_{miss} with the bound $x^{-g\beta}$, and provides a way to balance throughput, delay and security.

B. Performance monitoring the cluster head nodes

Fig. 8 shows the evolution of probability $(P_c)^X$ of the cluster head being compromised with the number of monitor nodes. It is not difficult to observe that $(P_c)^X$ reduces significantly as the number of monitor nodes increases. Furthermore, for an attacker, the head compromised probability should not be less than $(P_c)^X$. Therefore, if a WSN is deployed in the battle area (i.e., $P_c = 0.7$) and θ is required to be less than 0.2, it is easy to derive $X = 5$ by Fig. 8.

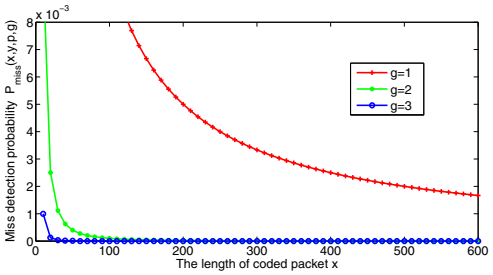

Fig. 6. Miss-detection probability versus the length of coded packet x for the different number of monitor nodes.

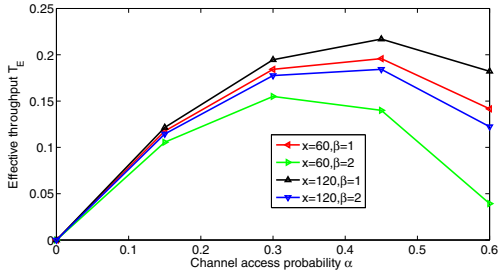

Fig. 7. Effective throughput versus channel access probability α with $y = x + 1 - \frac{\beta \ln x}{p}$.

C. Comparison of intrusion detection algorithm

In this part of the simulation, we compare the average energy consumption of CH node by our intrusion detection algorithm and that by the detection algorithm with one monitor node and without clustering. Fig. 9 shows that the average energy consumption of the detection algorithm compared is faster than that of our intrusion detection algorithm. Fig. 10 shows that the number of alive sensor nodes of our intrusion detection algorithm is much higher than that of the detection algorithm compared. This is because the detection algorithm compared does not consider the shortcoming of LEACH and the fact that each node has different workload. Hence, the nodes that are overused will fail fast. In contrast, sensor nodes with our intrusion detection algorithm have more longer lifetime, which is because we consider the residual energy of nodes.

VI. Conclusions

In this paper, we have studied the problem of clone nodes detection in wireless sensor networks. We introduce multiple monitor nodes into the detection process, where monitor nodes

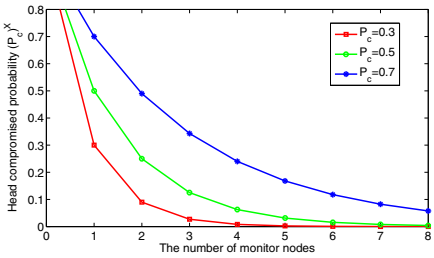

Fig. 8. Head compromised probability versus the number of monitor nodes with P_c.

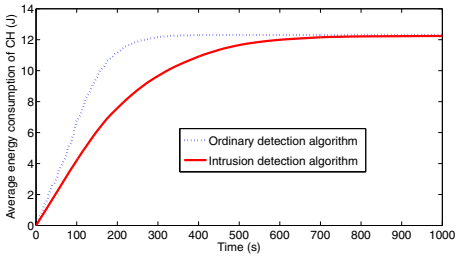

Fig. 9. Average energy consumption of CH node versus time.

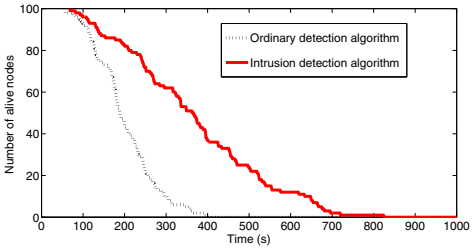

Fig. 10. Number of alive nodes versus time.

can observe the data transmission of all the nodes and the behavior of head clusters. By choosing the encoder function properly, we show that an attacker will be detected with high probability, and that the effective throughput provided by the proposed detection algorithm can arbitrarily approach the optimum. Besides, we propose an improved cluster protocol to cluster the network. The proposed protocol can improve the detection efficiency of the network and reduce the detection time. Meanwhile, the infected areas can be quickly isolated by our cluster protocol.

There are several challenging issues in our future work. First, we need to take into account how to detect the attack in the presence of multiple colluding adversaries. Second, we need to further study how the efficiency of detection can be affected when the monitor nodes have been captured.

VII. Acknowledgments

This work was supported by the Fundamental Research Funds for the Central Universities (XDJK2013C094, XDJK2013A018, 2362014XK12), the National Natural Science Foundation of China (No. 61170248, 61373179, 61373178, 61402381) and Science and Technology Leading Talent Promotion Project of Chongqing (cstc2013kjrc-ljrccj40001).

References

[1] S. Guo, C. Wang, and Y. Yang, "Joint mobile data gathering and energy provisioning in wireless rechargeable sensor networks," *IEEE Transactions on Mobile Computing*, pp. 2836–2852, 2014.

[2] C. Wang, J. Li, F. Ye, and Y. Yang, "Netwrap: An ndn based real-time wireless recharging framework for wireless sensor networks," *IEEE Transactions on Mobile Computing*, pp. 1283–1297, 2014.

[3] M. Ma, Y. Yang, and M. Zhao, "Tour planning for mobile data-gathering mechanisms in wireless sensor networks," *IEEE Transactions on Vehicular Technology*, pp. 1472–1483, 2013.

[4] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, Aug 2002.

[5] A. Becher, Z. Benenson, and M. Dornseif, "Tampering with motes: Real-world physical attacks on wireless sensor networks," *Security in Pervasive Computing*, vol. 3934, pp. 104–118, 2006.

[6] B. Parno, A. Perrig, and V. Gligor, "Distributed detection of node replication attacks in sensor networks," *Proc. IEEE Symposium on Security and Privacy*, pp. 49–63, May 2005.

[7] F. Liu, X. Cheng, and D. Chen, "Insider attacker detection in wireless sensor networks," *Proc. 26th IEEE International Conference on Computer Communications.*, pp. 1937–1945, May 2007.

[8] S. Umrao, D. Verma, and A. Tripathi, "Detection and mitigation of node replication with pulse delay attacks in wireless sensor network: A survey," *Proc. 2013 IEEE International Conference in MOOC Innovation and Technology in Education*, pp. 390–392, Dec 2013.

[9] Y. Zeng, J. Cao, S. Zhang, S. Guo, and L. Xie, "Random-walk based approach to detect clone attacks in wireless sensor networks," *Proc. IEEE Journal on Selected Areas in Communications*, vol. 28, no. 5, pp. 677–691, June 2010.

[10] Z. Li and G. Gong, "Randomly directed exploration: An efficient node clone detection protocol in wireless sensor networks," *Proc. 6th IEEE International Conference on Mobile Adhoc and Sensor Systems*, pp. 1030–1035, Oct 2009.

[11] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," *Proc. 9th ACM Conference on Computer and Communications Security*, pp. 41–47, 2002.

[12] A. P. R. da Silva, M. H. T. Martins, B. P. S. Rocha, A. A. F. Loureiro, L. B. Ruiz, and H. C. Wong, "Decentralized intrusion detection in wireless sensor networks," *Proc. 1th Inthernational Workshop on Quality of Service & Secutity in Wireless and Mobile Networks*, pp. 16–23, 2005.

[13] H. Choi, S. Zhu, and T. r. La Porta, "Set: Detecting node clones in sensor networks," *Proc. 3th International Conference on Security and Privacy in Communications Networks and the Workshops*, pp. 341–350, Sept 2007.

[14] K. Xing, F. Liu, X. Cheng, and D. Du, "Real-time detection of clone attacks in wireless sensor networks," *Proc. 28th International Conference on Distributed Computing Systems*, pp. 3–10, June 2008.

[15] X. Wang and J. Wong, "An end-to-end detection of wormhole attack in wireless ad-hoc networks," *Proc. 31th Annual IEEE International Conference on Computer Software and Applications*, vol. 1, pp. 39–48, July 2007.

[16] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," *Proc.6th Annual International Conference on Mobile Computing and Networking*, pp. 255–265, 2000.

[17] T. Ghosh, N. Pissinou, and K. Makki, "Collaborative trust-based secure routing against colluding malicious nodes in multi-hop ad hoc networks," *Proc.29th Annual IEEE International Conference on Local Computer Networks*, pp. 224–231, Nov 2004.

[18] S. Buchegger and J.-Y. Le Boudec, "Performance analysis of the confidant protocol," *Proc.3th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pp. 226–236, 2002.

[19] C. Zouridaki, B. L. Mark, M. Hejmo, and R. K. Thomas, "A quantitative trust establishment framework for reliable data packet delivery in manets," *Proc.3th ACM Workshop on Security of Ad Hoc and Sensor Networks*, pp. 1–10, 2005.

[20] S. Ganeriwal and M. B. Srivastava, "Reputation-based framework for high integrity sensor networks," *Proc.2th ACM Workshop on Security of Ad Hoc and Sensor Networks*, pp. 66–77, 2004.

[21] T. Zia, "Reputation-based trust management in wireless sensor networks," *Proc.International Conference on Sensor Networks and Information*, pp. 163–166, Dec 2008.

[22] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *Proc. IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660–670, Oct 2002.

[23] S. Bandyopadhyay and E. Coyle, "Minimizing communication costs in hierarchically clustered networks of wireless sensors," *IEEE Wireless Communications and Networking*, vol. 2, pp. 1274–1279 vol.2, March 2003.

[24] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," *Proc.33th Annual Hawaii International Conference on System Sciences*, Jan 2000.

[25] H. Balli, X. Yan, and Z. Zhang, "On randomized linear network codes and their error correction capabilities," *Proc. IEEE Transactions on Information Theory*, vol. 55, no. 7, pp. 3148–3160, July 2009.

[26] C.-K. Ngai, R. Yeung, and Z. Zhang, "Network generalized hamming weight," *Proc. Network Coding Workshop on Theory, and Applications*, pp. 48–53, June 2009.