

# Randomly Directed Exploration: An Efficient Node Clone Detection Protocol in Wireless Sensor Networks

Zhijun Li and Guang Gong

Department of Electrical and Computer Engineering

University of Waterloo, Waterloo, Ontario, Canada

Email: leezj@engmail.uwaterloo.ca and ggong@calliope.uwaterloo.ca

## Abstract

*Node clone attack, that is, the attempt by an adversary to add one or more nodes to the network by cloning captured nodes, imposes a severe threat to wireless sensor networks. Several distributed detection protocols have been proposed against this attack. However, all of them rely on too strong assumptions and cannot be efficiently applied to most of sensor networks. In this paper, we propose an innovative randomly directed exploration protocol to detect the node clone. Each node need only know its neighbors' information, and then collaborates to forward claiming messages, trying to find out clone. No any specific routing protocols or infrastructures are demanded in the proposed protocol. Therefore, it is highly practical in the general sensor network applications. In addition, the memory requirement of the protocol is almost optimal. Furthermore, the protocol consumes relatively low communication overload, which is not inferior to any previous schemes. The simulation results show that the protocol can achieve high detection probability. Overall, the proposed protocol outweighs previous approaches in terms of practicability and performance.*

## 1. Introduction

Wireless sensor networks (WSNs) are innovative wireless networks consisting of a large number of low-cost, recourse-constrained, distributed commodity sensor nodes that collaboratively collect information. Sensor nodes deployed in hostile environments are vulnerable to capture and compromise. Because of production cost limitation, sensor nodes are short of tamper-resistance hardware components. An adversary can capture a few nodes, extracts code and all secret credentials, and uses those materials to clone many nodes out of off-the-shelf sensor hardware. Then those nodes that seem legitimate are able to join the network and cause severe damages. For example, the cloned nodes can occupy strategic positions and cooperatively corrupt the collected information. They can even let the adversary control the whole network. Furthermore, the node clone would exasperate most of inside attacks against sensor networks.

Due to reliability and balance considerations, the distributed detection approaches against the node clone attack are more suitable for most sensor network applications. Several distributed schemes [1]–[3] have been proposed. However, some protocols rely on an implicit assumption that every node knows the existence of all other nodes and require an underlying efficient node-to-node unicast routing protocol, while other protocols based on geographic hash tables demand sensor nodes to realize the outline of the network geography. Those assumptions are too strong to hold in many sensor networks. In addition, the performance of those protocols is not satisfactory.

Generally, energy is the scarcest resource in wireless sensor networks since sensor nodes are powered by batteries and recharging is difficult in many circumstances. Communication consumes at least one order of magnitude power than any of other operations [4]. Therefore, we evaluate the communication overload as the main protocol performance. For simplicity, we use the average number of sent messages per node to measure a protocol's communication cost. Furthermore, sensor nodes are only equipped with a limited amount of memory; thus any schemes requiring high storage would be considered to be impractical. The memory requirement would be another performance metric .

In this paper, we present a novel node clone detection protocol—randomly directed exploration. This protocol does not call for any unrealistic assumptions. Each node only needs to know its neighbor nodes. During the detection procedure, nodes issue claiming messages containing neighbor-list with a maximum hop limit to randomly selected neighbors. The previous transmission of a claiming message forms a direction, and then the intermediate node tries to follow the direction to forward the message. During forwarding messages, the intermediate nodes explore the claiming messages for node clone detection. In such a simple way, the proposed protocol can efficiently detect node clone in the dense sensor networks. In addition, the protocol consumes almost minimum memory during detection, and communication payload is satisfactory. It can scale to large configurations. We implement the protocol in the OMNet++ simulation framework [5], and the simulation results show that our protocol presents good performance and holds

strong resistance against active adversaries.

The rest of the paper is organized as follows. First, the previous approaches against the node clone attack are introduced in Section 2. Then we define the network model and adversary model in Section 3. Afterwards, we detail our proposed detection protocol and analyze its performance in Section 4. The simulation results are provided in Section 5. Finally, we conclude our work in Section 6.

## 2. Previous Work

Protocols which can prevent clone nodes from joining the sensor networks look like better than detection schemes. Bekara and Laurent-Maknavicius [6] described a node clone prevention protocol based on the initial trust model. In [7], Zhang et al. proposed the use of location-based keys to thwart and defend against several attacks, including the node clone. However, those prevention approaches are highly application-specific, and cannot be applied to general sensor networks.

In a simplest centralized detection approach, each node sends a list of its neighbor nodes and their claimed locations to a base station, which is responsible for detecting the node clone. SET, proposed by Choi, Zhu, and Porta [8], manages to reduce the communication cost of the preceding approach by computing set operations of exclusive subsets in the network. SET first launches an exclusive subset maximal independent set (ESMIS) algorithm which constructs exclusive unit subsets among one-hop neighbors in a distributed way. Consequently, each node is located in one and only one disjointed subset which is controlled by a randomly decided leader. Then those subsets, in the basic scheme, are transmitted by leaders to the base station such that it can construct all nodes locations and detect clones. Since the subset division procedure eliminates redundancy in node location reports, SET lowers the communication cost. However, in order to prevent malicious nodes in the ESMIS algorithm, an authenticated subset covering protocol has to be performed, which increases the communication overload and the overall complexity.

Brooks *et al.* [9] proposed a cloned-key detection protocol in the context of random key predistribution [10]. The basic idea is that the keys employed according to the random key predistribution scheme should follow a certain pattern, and those keys whose usage exceeds a threshold can be judged to be cloned. In the protocol, every node reports its keys to a base station and then the base station uses an abnormality-based intrusion-detection-like statistical technique to detect cloned keys. A general issue in this kind of approaches is the high false negative and positive rates. Furthermore, the authors do not address how to make malicious nodes to report their keys honestly. Without such an assurance, the detection is meaningless.

As mentioned in [1], the centralized approaches may create a single-point of failure. Besides, there might not exist a powerful central point in some applications. Furthermore, one central point will be an obvious attacking target for adversaries, and the nodes surrounding the central point would suffer an undue communication burden, which may shorten the network's life expectancy. In many cases, a distributed and balanced detection scheme is more desirable.

Node-to-node broadcasting [1] is a practical and effective method to distributively detect the node clone. The main problem in this approach is the high communication overload. Parno, Perrig, and Gligor [1] provided two probabilistic witness-based protocols. Randomized multicast scheme distributes node location information to randomly-selected witnesses, exploiting the birthday paradox to detect cloned nodes, while line-selected multicast scheme uses the topology of the network to detect replication, that is, in addition to witness nodes, the nodes within the multicast path check the node clone. The storage consumptions in the both schemes are very high. Besides, the communication cost in the randomized multicast is similar to that in the node-to-node broadcasting. Thus there is no benefit to use the randomized multicast. For the procedure of choosing random witness, both schemes imply that every node is aware of all other nodes' existence, and the so-called multicast requires an underlying node-to-node routing protocol. Those requirements greatly constraints their applicability in sensor networks.

Geographic Hash Tables (GHTs) [11] map data keys into geographical coordinations, creating distributed data-centric storage systems. Based on GHTs, Zhu *et al.* [2] proposed a localized multicast to detect the node clone. In the system, the witness nodes for an examined node are selected from nodes that are located within a geographical limited region (named cell) which is determined by a GHT hash result of the examined node's identification. They presented two variants of localized multicast: single deterministic cell, in which only one unique cell is determined for a node, and parallel multiple probabilistic cell, in which the location claim is mapped and forwarded to multiple deterministic cells with various probabilities. Conti *et al.* [3] proposed another GHT-based clone detection approach. The efficiency of GHT-based approaches is highly related to the network topology. Essentially, those approaches rely on the nodes awareness of the general deployed geography of sensor networks. This prerequisite may hold in some circumstances, but cannot be guaranteed in the general cases.

## 3. Network and Adversary Models

To demonstrate our distributed protocol, we define the network and adversary models used in this paper.

### 3.1. Network Model

We consider a densely deployed sensor network consisting of  $N$  resource-constrained sensor nodes with the average node degree  $d$  (the number of neighbors). If  $d$  is small, then the node-to-node broadcasting will be sufficient to detect node clone efficiently. According to the traditional sensor network setting, the node number  $N$  can be huge. Therefore, a practical protocol should scale to large network sizes. Sensor nodes operate without supervision at most of time, and they can function correctly in a dynamic network, where new nodes are added, or old nodes disappear. Sensor nodes are only aware of its neighbor nodes.

We assume, like the previous distributed detection approaches [1], [3], [11], that an identity-based public-key cryptography [12] is implemented in the system. Before deployment, each legitimate node is assigned with a unique ID and a corresponding private key by a trusted third party. The public key of a node is its ID, which is the essence of identity-base cryptosystem. This implies a strong entity authentication scheme for sensor nodes. Moreover, recipients can verify messages signed by a node using the identity-based key. Let  $K_\alpha$  and  $K_\alpha^{-1}$  denote the public and private keys of node  $\alpha$  respectively, and  $\{M\}_{K_\alpha^{-1}}$  represent the signature of  $M$  signed by node  $\alpha$ . Compared to traditional public key cryptosystems, which can satisfy the requirements of our system as well, the identity-based systems alleviate the heavy burden of public key certificates. The applicable implementation of public-key cryptosystems in typical sensor nodes platform have been addressed in [13]–[15].

We also assume that every sensor node can determine its position  $L$  via a secure localization mechanism. A number of those mechanisms have been proposed, which can be referred to in [16]. During a round of node clone detection, we suppose the sensor network to be stationary; so a collision of locations for one node ID would be concluded to be a clone.

### 3.2. Adversary Model

We consider a threat model in which sensor nodes are deployed in a hostile environment and are subject to being compromised by an adversary, but the adversary is only capable of controlling a small portion of sensor nodes. If the adversary takes control of the majority of nodes, all security mechanisms will eventually fail. The adversary can use the compromised nodes to clone many nodes and deploy the replicas in places that are decided intelligently.

The adversary is aware of the detection protocol and manages to conceal the existence of clone. In our settings, the adversary is allowed to interfere with the detection scheme in three ways. First, the cloned nodes may not participate the regular detection procedures. Second, the cloned nodes may drop or manipulate the reporting messages

which they forward. Lastly, the adversary can capture some key nodes as a countermeasure against the detection.

## 4. Proposed Protocol

We notice that the node-to-node broadcasting scheme is the most practical one among the previous distributed detection approaches. It does not require any additional assumption—every node just simply broadcasts its neighbor-list to other nodes. In addition, each node need only buffer its own neighbor-list, so the memory requirement is pretty low, except for the additional memory cost for preventing from receiving the same broadcasting messages. However, it incurs the heavy transmission cost. Overall,  $O(N^2)$  messages are sent during one detection procedure. We observe that if the sensor network is dense, most of node-of-node broadcasting detection messages for detecting clone are redundant. Assume that there are two clone nodes in the network, each of which has  $d$  integrity neighbors. To detect the clone successfully, it is sufficient that one of the  $d$  neighbors of one cloned node receives a claiming message from one of the  $d$  neighbor of the other cloned node. In this sense, instead of broadcast, anycast [17] seems a good alternative for node clone detection. However, we cannot afford an infrastructure for current anycast protocols in the sensor networks. Instead, we present the innovative randomly directed exploration protocol to achieve the efficient node clone detection.

### 4.1. Protocol Description

Upon deployed, every node notifies all neighbors about its ID and location. Then each node creates its own neighbor-list including the neighbors IDs and locations. This neighbor-list will be used for the clone detection and for assisting message routing.

The global parameter set used in the directed exploration is  $(type, ttl, c)$ , where *type* is the routing type, which determines how to select the next node during forwarding a claiming message, *ttl* is the time to live—the maximum hop count for a claiming message, and *c* is the average number of claiming messages per node. Note that *c* is a real number instead of an integer.

During one detection procedure, each node starts to generate  $c$  claiming messages. For each claiming message, the node transmits it to a *randomly* selected neighbor. For intermediate nodes forwarding claiming messages, the way how to determine the next node constitutes the *directed exploration* technique. Since every node has information about its own location and all neighbors locations, it can easily calculate all neighbors angels. When node  $\alpha$  receives message  $M$  from previous node  $\beta$ , the next destination node  $\gamma$  for message  $M$  is chose by node  $\alpha$  such that  $\gamma$ 's angle is most opposite to  $\beta$ 's angle, that is,  $\gamma.angle$  is closest to  $(\beta.angle + \pi)$ . Forwarded by all intermediate nodes in

---

**Algorithm 1** *handlemessage*( $M_\alpha$ )

---

```
1: verify the signature of  $M_\alpha$ 
2: if found clone then
3:   broadcast the evidence;
4:  $t_{tl} \leftarrow t_{tl} - 1$ 
5: if  $t_{tl} \leq 0$  then
6:   discard  $M_\alpha$ 
7: else
8:    $nextnode \leftarrow getnextnode(M_\alpha)$ 
9:   if  $nextnode = \text{NIL}$  then
10:    discard  $M_\alpha$ 
11:  else
12:    forward  $M_\alpha$  to  $nextnode$ 
```

---

this way, the message would travel the network roughly as a line. The message's  $t_{tl}$  will be decreased by 1 each time it is forwarded by a node. Once  $t_{tl} = 0$ , the message will be discarded. We present two types of routings for the randomly directed exploration protocol: Type 1 and Type 2. For Routing Type 1, a message will be discarded only when its  $t_{tl}$  is decreased to 0. Type 1 is suitable for all kinds of sensor network topologies. By the contrary, Type 2 takes the network topology into consideration to reduce the communication cost. In many sensor network applications, nodes are randomly deployed, and there exist some outside borders of network. When reaching some border in the network, the claiming message can be directly discarded. Specifically, an *anglerange* is defined in Type 2. When no node is found in the angle range, the current node would conclude that the message has reached a border, and thus throw it away.

The claiming message by node  $\alpha$  is defined as follows.

$$M_\alpha = t_{tl}, ID_\alpha, L_\alpha, NeighborList_\alpha, \{ID_\alpha, L_\alpha, NeighborList_\alpha\}_{K_\alpha^{-1}}$$

Since  $t_{tl}$  will be altered by intermediate nodes during transformation, it should not be authenticated.

After receiving message  $M_\alpha$ , node  $\beta$  calls *handlemessage*( $M_\alpha$ ), which is described by pseudo-code in Algorithm 1, to process the message.

During handling a message, the node compares its own neighbor-list with the neighbor-list in the message, checking if there is a clone. If detecting a clone, node  $\beta$  would broadcast an evidence message,  $M_{evidence} = M_\alpha, M_\beta$ , to notify the whole network. Notice that messages  $M_\alpha$  and  $M_\beta$  are authenticated by nodes  $\alpha$  and  $\beta$  respectively. Therefore, all integrity nodes can verify the evidence message and stop communicating with the cloned nodes. To prevent cloned nodes from joining the network in the future, a revocation list of cloned nodes IDs may be maintained by every node.

Algorithm 2 describes how to select the next node for a message, according to the routing type setting.

---

**Algorithm 2** *getnextnode*( $M_\alpha$ )

---

```
1:  $idealangle \leftarrow previousnode.angle + \pi$ 
2:  $nextnode \leftarrow$  node whose angle is closest to  $idealangle$ 
3: if  $nextnode = previousnode$  then
4:   return NIL
5: if  $type = 1$  then
6:   return  $nextnode$ 
7: else if  $type = 2$  then
8:   if  $nextnode.angle \in [idealangle - anglerange, idealangle + anglerange]$  then
9:     return  $nextnode$ 
10:  else
11:    return NIL
12: else
13:   return NIL
```

---

## 4.2. Analysis

Apparently, the randomly directed exploration protocol is highly memory-efficient. It does not rely on broadcasting. As a result, no additional memory is required to suppress broadcasting flood. The proposed protocol does not demand the intermediate nodes to buffer claiming messages, overcoming the main advantage of line-selected multicast scheme [1]. All memory requirement lies to the neighbor-list. In fact, the neighbor-list is a necessary component for all distributed detection approaches. Therefore, the proposed protocol consumes almost minimum memory.

The communication cost of the randomly directed exploration depends on the routing parameter settings. For Routing Type 1, the communication cost will be fixed,  $c \times t_{tl}$  messages sent by per node. For Routing Type 2, within appropriate network topologies, the communication cost will be a portion of  $c \times t_{tl}$ . For a dense sensor network,  $c = 1.0$  might be a good choice, and  $t_{tl} = \sqrt{N}$  would be sufficient for the maximum hop of message to go across the network. The choice of  $\sqrt{N}$  is also used in the [1]. Therefore, the upper-bound of communication cost in the randomly directed exploration protocol is  $O(\sqrt{N})$ . The performance comparison between the randomly directed exploration protocol and the existing distributed detection protocols is shown in Table 1.

Table 1: Performance Comparison

Protocol	Comm. Cost	Memory Cost
Node-To-Network Broadcasting [1]	$O(N)$	$O(d)$
Randomized Multicast [1]	$O(N)$	$O(\sqrt{N})$
Line-Selected Multicast [1]	$O(\sqrt{N})$	$O(\sqrt{N})$
Randomized, Efficient, and Distributed [3]	$O(\sqrt{N})$	$O(d\sqrt{N})$
Single Deterministic Cell [2]	$O(\sqrt{N})$	$< O(\sqrt{N})$
Parallel Multiple Probabilistic Cells [2]	$O(\sqrt{N})$	$< O(\sqrt{N})$
Randomly Directed Exploration	$O(\sqrt{N})$	$O(d)$

The security of the proposed protocol relies on the identity-based public-key system, which guarantees the authentication of node's identity and message integrity. The adversary cannot fake clone nodes' ID. A cloned node cannot lie to its neighbors about its location since the faked location would be far deviated from the communication range of the neighbors. On the other hand, the usage of public-key system might be a main burden for the protocols practicality in some sensor network applications. As discussed in [1], the public-key system can be replaced by symmetric primitives with some tradeoffs.

Limited by the secure message authentication, adversaries cannot manipulate the legitimate claiming messages. The best thing that cloned nodes can perform against detection is to discard claiming messages passing through them. If the number of cloned nodes  $b$  is very small, the effect of such action is negligible. If adversaries try to employ more cloned nodes, it will greatly increase the detection probability as there are  $O(bd)$  nodes that would issue claiming messages including clone clue, and one of the  $O(bd)$  nodes receiving other's clue would be enough to detect the clone.

## 5. Simulations

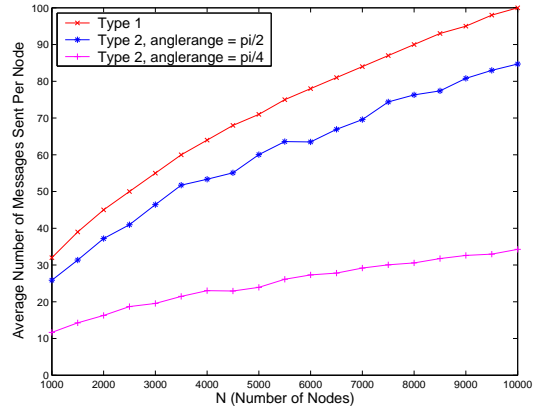
To evaluate the performance, we implement our protocol and run simulations in the OMNeT++ framework [5], because of its good scalability and high efficiency, which facilitates simulating large scale of networks.

The network scenario used in our simulation is the *Unit-Disc Graph*, in which nodes are uniformly deployed in a  $500 \times 500$  square and nodes follow the standard unit-disc bidirectional communication model. We adjust the node communication range such that the average node degree keeps the approximate  $d$ . This kind of simulation scenario has been used in [1] and many other literatures. In the simulations, we choose the fixed average node degree  $d = 20$ . We use three settings to test the protocol performance: Routing Type 1, Routing Type 2 with  $anglerange = \frac{\pi}{2}$ , and Routing Type 2 with  $anglerange = \frac{\pi}{4}$ .

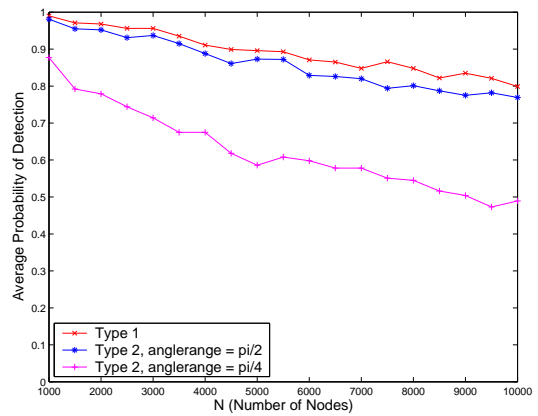
### 5.1. Performance in Different Network Sizes

We design and run Experiment One to measure the randomly directed exploration protocol's performance within the different network size. We launch 19 runs of simulations, in which the network size  $N$  ranges from 1000 to 10000, with a step of 500. Every run performs 100 rounds of detection, in each of which two nodes are randomly chosen to set the same ID, that is, those two are cloned nodes. We set  $tll = \lceil \sqrt{N} \rceil$  and  $c = 1.0$ . We repeat running experiments 10 times using different random seeds. In Experiment One, we make the cloned nodes to obey the detection protocol.

We measure the communication cost by the average number of sent messages per node in one round, which



(a) Communication Cost



(b) Probability of Detection Clone

Figure 1: Protocol Performance in Different Network Sizes, where  $tll = \sqrt{N}$ ,  $c = 1.0$

is depicted in Figure 1a. In the graph, the communication overloads in the simulations of Type 1 are exactly  $\lceil \sqrt{N} \rceil$ . On the contrary, Type 2 incurs less transmission overload. The communication costs of Routing Type 2 with  $anglerange = \frac{\pi}{2}$  and  $anglerange = \frac{\pi}{4}$  are roughly 83.4% and 35.0% respectively of that of Routing Type 1.

Figure 1b depicts the average successful probability of detecting clone. Both Type 1 and Type 2 with  $anglerange = \frac{\pi}{2}$  achieve high detection probability. According to Figure 1, Routing Type 2 with  $anglerange = \frac{\pi}{2}$  is a proper choice in the *Unit-Disc Graph* network model. However, we have to mention that Type 1 adapts better to irregular network topologies than Type 2. Due to space limitations, we do not provide the simulation results in different irregular topologies.

### 5.2. Resilience against Active Adversaries

We develop Experiment Two to evaluate the protocol's performance if the cloned nodes are allowed to discard messages. We test with one network size  $N = 1000$ , and the

cloned node number  $b$  varies from 2 to 100. For each run, we repeat 100 rounds of node detection, in each of which  $b$  nodes are randomly chosen as clones. The result shows that the protocol achieves 100% detection probability for ( $b \geq 3$  in Type 1 and Type 2 with  $\text{anglerange} = \frac{\pi}{2}$ ), and ( $b \geq 4$  in Type 2 with  $\text{anglerange} = \frac{\pi}{4}$ ). As a matter of fact, if there are several cloned nodes using a same ID employed in the network, our protocol would almost certainly detect the clone. We use the same setting for Experiment One except choosing three randomly nodes as clones that are allowed to discard messages. The simulation for Type 2 with  $\text{anglerange} = \frac{\pi}{2}$  shows at least 98% detection probability for all network sizes between 1000 and 10000.

## 6. Conclusion

We present the randomly directed exploration protocol for distributed node clone detection in this paper. Using the elegant, efficient directed-forwarding technique along with the initial randomness, the protocol achieves high detection probability with remarkable communication and memory cost. The proposed protocol is more practical than the previous approach and can be applied to most of sensor networks.

## Acknowledgment

The research is supported by NSERC Strategic Project Grants.

## References

- [1] B. Parno, A. Perrig, and V. Gligor, "Distributed Detection of Node Replication Attacks in Sensor Networks," in *Proceedings of the IEEE Symposium on Security and Privacy*, 2005, pp. 49 – 63.
- [2] B. Zhu, V. G. K. Addada, S. Setia, S. Jajodia, and S. Roy, "Efficient Distributed Detection of Node Replication Attacks in Sensor Networks," in *Twenty-Third Annual Computer Security Applications Conference (ACSAC 2007)*, 2007, pp. 257–267.
- [3] M. Conti, R. D. Pietro, L. V. Mancini, and A. Mei, "A randomized, efficient, and distributed protocol for the detection of node replication attacks in wireless sensor networks," in *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*. Montreal, Quebec, Canada: ACM, 2007.
- [4] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, 2002.
- [5] "OMNeT++ Community Site," <http://www.omnetpp.org/>.
- [6] C. Bekara and M. Laurent-Maknavicius, "A New Protocol for Securing Wireless Sensor Networks against Nodes Replication Attacks," in *Third IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMOB 2007)*, 2007, pp. 59–59.
- [7] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "Location-based compromise-tolerant security mechanisms for wireless sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 247–260, 2006.
- [8] H. Choi, S. Zhu, and T. F. La Porta, "SET: Detecting node clones in sensor networks," in *Third International Conference on Security and Privacy in Communications Networks and the Workshops (SecureComm 2007)*, 2007, pp. 341–350.
- [9] R. Brooks, P. Y. Govindaraju, M. Pirretti, N. Vijaykrishnan, and M. T. Kandemir, "On the Detection of Clones in Sensor Networks Using Random Key Predistribution," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 37, no. 6, pp. 1246–1258, 2007.
- [10] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proceedings of the 9th ACM conference on Computer and Communications Security*, Washington, DC, USA, 2002, pp. 41–47.
- [11] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker, "GHT: a geographic hash table for data-centric storage," in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications (WSNS)*. Atlanta, Georgia, USA: ACM, 2002.
- [12] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proceedings of CRYPTO 84 on Advances in cryptology, Lecture Notes in Computer Science 196*. Springer-Verlag, 1984, pp. 47–53.
- [13] A. Liu and P. Ning, "TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks," in *International Conference on Information Processing in Sensor Networks (IPSN '08)*, 2008, pp. 245–256.
- [14] P. Szczechowiak, L. Oliveira, M. Scott, M. Collier, and R. Dahab, "NanoECC: Testing the Limits of Elliptic Curve Cryptography in Sensor Networks," in *Wireless sensor networks*. LNCS 4913, 2008, pp. 305–320.
- [15] L. B. Oliveira, M. Scott, J. Lopez, and R. Dahab, "TinyPBC: Pairings for authenticated identity-based non-interactive key distribution in sensor networks," in *Networked Sensing Systems, 2008. INSS 2008. 5th International Conference on*, 2008, pp. 173–180.
- [16] R. Poovendran, C. Wang, and S. Roy, *Secure Localization and Time Synchronization for Wireless Sensor and Ad Hoc Networks*. Springer Verlag, 2007.
- [17] D. Xuan, W. Jia, W. Zhao, and H. Zhu, "A routing protocol for anycast messages," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 11, no. 6, pp. 571–588, 2000.