Clone Node Detection in Wireless Sensor Networks

Mr.R.Rajaguru[1], Mr.P.Ramachandran[2]
[1]PG Student, [2]Professor
[1 2] Department of CSE, [1 2] Sri Ramanujar Engineering College, Tamil Nadu, India.

***Abstract*--**WSN (Wireless Sensor Network) is spatially distributed sensor that is used to monitor the physical conditions as well as environmental conditions such as sound, temperature, pressure to pass their information through the network to main place. Wireless Sensor Networks mainly involve in consumer and industrial such as security, control and several distributed protocols have been proposed to detect this attack in wireless sensor networks. In this paper, we propose two novel node clone detection protocols with different tradeoffs on network conditions and performance. The first one is based on a distributed hash table (DHT), by which a fully decentralized, key-based caching and checking system is constructed to catch cloned nodes effectively. Although the DHT-based protocol incurs similar communication cost as previous approaches, it may be considered a little high for some scenarios. The protocol performance on efficient storage and high security level is theoretically deducted through a probability model, and the resulting equations, with necessary adjustments for real application, are supported by the simulations The second distributed detection protocol, named randomly directed exploration, presents good communication performance for dense sensor networks, by a probabilistic directed forwarding technique along with random initial direction and border determination .Distributed detection, distributed hash table, node clone attack, randomly directed exploration, wireless sensor networks (WSNs).

Keywords: *Distributed detection, distributed hash table, node clone attack, randomly directed exploration, wireless sensor networks (WSNs).*

I. INTRODUCTION

A wireless sensor network (WSN) is a computer network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different locations. The development of wireless sensor networks was originally motivated by military applications such as battlefield surveillance. However, wireless sensor networks are now used in many civilian application areas, including environment and habitat monitoring, healthcare applications, home automation, and traffic control.

The number of sensor nodes in a sensor network can be several orders of magnitude higher than the nodes in an ad chord network. Sensor nodes mainly use broadcast communication paradigm whereas most ad chord networks are based on end-to-end communications. In addition to one or more sensors, each node in a sensor network is typically equipped with a radio transceiver or other wireless communications device, a small microcontroller, and an energy source, usually a battery. The size a single sensor node can vary from shoebox sized nodes down to devices the size of grain of dust. The cost of sensor nodes is similarly variable, ranging from hundreds of dollars to a few cents, depending on the size of the sensor network and the complexity required of individual sensor nodes.

Randomly directed exploration, is intended to provide highly efficient communication performance with adequate detection probability for dense sensor networks. In the protocol, initially nodes send claiming messages containing a neighbor -list along with a maximum hop limit to randomly selected neighbors; then, the subsequent message transmission is regulated by a probabilistic directed technique to approximately maintain a line property through the network as well as to incur sufficient randomness for better performance on communication and resilience against adversary .The first proposal is based on a distributed hash table (DHT) [2], In addition, border determination mechanism is employed to further reduce communication payload. During forwarding, intermediate nodes explore claiming messages for node clone detection. By design, this protocol consumes almost minimal memory, and the simulations show that it outperforms all other detection protocols in terms of communication cost, while the detection probability is satisfactory.

II. BASIC WIRELESS SENSOR NETWORK TECHNOLOGY

The WSN has many features helping the technology to be deployed in real life application as soon as possible even though these feature differ depending on the technology, here is a list of them, The network architecture depends on the application deploying WSN. For example, some nodes are connected directly to the sink without passing through other nodes (1-hop layer). Other layers might go through other nodes to forward the data to the sink. Fig.1.1 shows these different layers. A very large number of nodes, often in the order of thousands Asymmetric flow of information, from sensor nodes to a command node Communications are triggered by events.

At each node there is a limited amount of energy which in many applications is impossible to replace Low cost, size, and weight per node. More use of broadcast communications instead of point-to-point Nodes do not have a global ID such as an IP number. The security, both physical and at the communication level, is more limited than conventional wireless networks.

III. CHORD NETWORK

In computing, Chord is a protocol and algorithm for a peer-to-peer distributed hash table. A distributed hash table stores key-value pairs by assigning keys to different computers (known as "nodes"); a node will store the values

for all the keys for which it is responsible. Chord specifies how keys are assigned to nodes, and how a node can discover the value for a given key by first locating the node responsible for that key.

IDs and keys are assigned an m bit identifier using consistent hashing. The SHA-1 algorithm is the base hashing function for consistent hashing. Consistent hashing is integral to the robustness and performance of Chord because both keys and IDs (IP addresses) are uniformly distributed and in the same identifier space. Consistent hashing is also necessary to let nodes join and leave the network without disruption. Using the Chord lookup protocol, node keys are arranged in a circle that has at most 2m nodes. The circle can have IDs keys ranging from 0 to 2m -1.

A. Stabilization

To ensure correct lookups, all successor pointers must be up to date. => stabilization protocol running periodically in the background. Updates finger tables and successor pointers. Stabilization protocol: Stabilize(): n asks its successor for its predecessor p and decides whether p should be n‘s successor instead (this is the case if p recently joined the system) [2]. Notify(): notifies n‘s successor of its existence, so it can change its predecessor to n

B. Consistent Hashing

Consistent hash function assigns each node and key an m-bit identifier.SHA-1 is used as a base hash function. A node’s identifier is defined by hashing the node’s IP address.A key identifier is produced by hashing the key (chord doesn’t define this. Depends on the application).

a. ID(node) = hash(IP, Port)

b. ID(key) = hash(key)

Each node has a successor and a predecessor. The successor to a node (or key) is the next node in the identifier circle in a clockwise direction. The predecessor is counter-clockwise. If there is a node for each possible ID, the successor of node 2 is node 3, and the predecessor of node 1 is node 0; however, normally there are holes in the sequence. For example, the successor of node 2 may be node 6 (and nodes from 3 to 7 will not exist), in this case, the predecessor of node 7 will be node 3. That happens because not all nodes of the Chord are actual nodes. Some of them may be virtual ones.

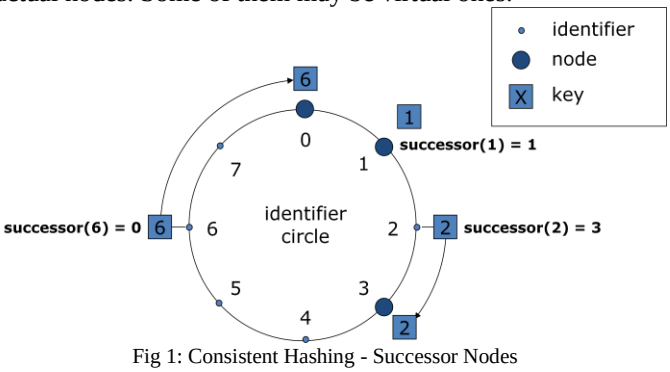


Fig 1: Consistent Hashing - Successor Nodes
(Ex: three sites/nodes at 0, 1, 3)

In an m-bit identifier space, there are 2m identifiers. Identifiers are ordered on an identifier circle modulo 2m. The identifier ring is called Chord ring.Key k is assigned to the first node whose identifier is equal to or follows (the identifier of) k in the identifier space. This node is the successor node of key k, denoted by successor(k). It is not uncommon to use the words "nodes" and "keys" to refer to these identifiers, rather than actual nodes or keys. A logical ring with positions numbered 0 to 2m -1 is formed among nodes. Key k is assigned to node successor(k), which is the node whose identifier is equal to or follows the identifier of k. If there are N nodes and K keys, then each node is responsible for roughly K/N keys. When a new node joins or leaves the network, responsibility for O(K/N) keys changes hands. Chord implements a faster search method. Chord requires each node to keep a "finger table" containing up to entries. The ith entry of node n will contain the address of successor(n=2i-1 mode2m).

With such a finger table, the number of nodes that must be contacted to find a successor in an N-node network is O(log N).

C. SYSTEM ARCHITECTURE,MODELS AND SECURITY REQUIREMENTS

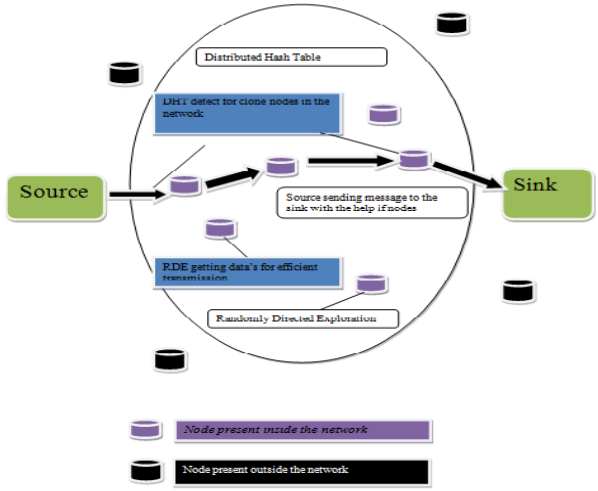


Fig. 2: System Architecture

System Architecture diagram describes the overall process between the distributed detection protocol is to make use of the DHT mechanism to form a decentralized caching and checking system that can effectively detect cloned nodes. Essentially, DHT enables sensor nodes to distributive construct an overlay network upon a physical sensor network and provides an efficient key-based routing within the overlay network. A message associated with a key will be transmitted through the overlay network to reach a destination node that is solely determined by the key; the source node does not need to specify or know which node a

message's destination is the DHT key-based routing takes care of transportation details by the message's key.

More importantly, messages with a same key will be stored in one destination node. Those facts build the foundation for our first detection protocol. As a beginning of a round of DHT-based clone detection, the initiator broadcasts the action message including a random seed. Then, every observer constructs a claiming message for each neighbor node, which is referred to as an examinee of the observer and the message, and sends the message with probability independently. The introduction of the claiming probability is intended to reduce the communication overwork in case of a high-node-degree network. In the protocol, a message's DHT key that determines its routing and destination is the hash value of concatenation of the seed and the examinee ID. By means of the DHT mechanism, a claiming message will eventually be transmitted to a deterministic destination node, which will cache the ID-location pair and check for node clone detection, acting as an inspector. In addition, some intermediate nodes also behave as inspectors to improve resilience against the adversary in an efficient way.

D. System Model and Start a round of detection

In this project consider a large-scale, homogeneous sensor network consisting of n resource-constrained sensor nodes. Shown in Fig.1,we consider a system consists of entities Analogous to previous distributed detection approaches, Assume that an identity-based public-key cryptography facility is available in the sensor network. Prior to deployment, each legitimate node is allocated 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 an identity-based cryptosystem. Consequently, no node can lie to others about its identity. Moreover, anyone is able to verify messages signed by a node using the identity based key. Let Kα and Kα-1 denote the public and private keys of node α , respectively, and {M} Kα-1 represent the signature of M signed by node α

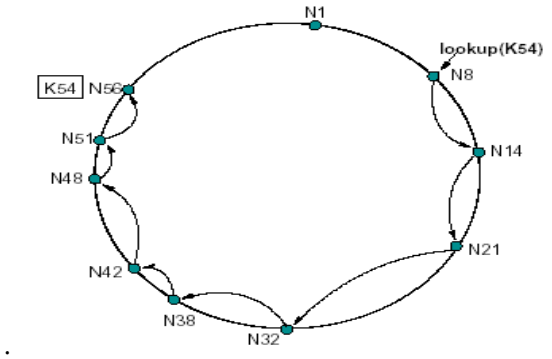


Fig 3: The path taken by a query from node 8 for key 54:

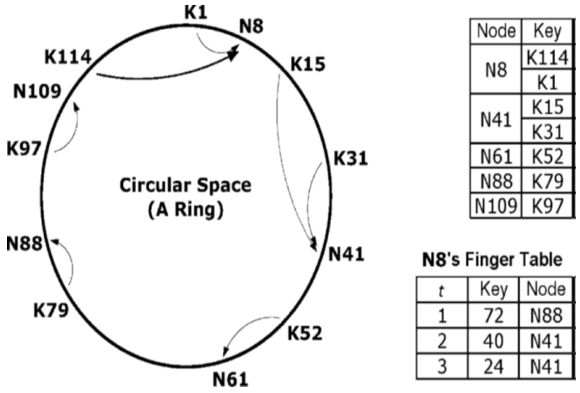
Based on the geographic hash table, which maps a key into a geographical coordination, Zhu et al.and Conti et al.pro-posed several clone detection schemes. In Addition also assume that every sensor node can determine its geographic location and current relative time via a secure localization protocol and a secure time synchronization scheme, respectively. There may or may not be a powerful base station in our modeled network, but there should exist a trusted role named initiator that is responsible for initiating a distributed detection procedure. Otherwise, an adversary can readily launch a denial-of-service (DoS) attack to the system by repeatedly mobilizing the sensor network to conduct the clone detection protocol and exhausting nodes energy. DHT - based detection protocol .

There are several different types of DHT proposals, such as CAN, Chord, and Pastry. Generally, CAN has least efficiency than others in terms of communication cost and scalability, and it is rarely employed in real systems More importantly, messages with a same key will be stored in one destination node. Those facts build the foundation for our first detection protocol. As a beginning of a round of DHT-based clone detection, the initiator broadcasts the action message including a random seed. The introduction of the claiming probability is intended to reduce the communication overwork in case of a high-node-degree network. In the protocol, a message's DHT key that determines its routing and destination is the hash value of concatenation of the seed and the examinee ID. By means of the DHT mechanism, a claiming message will eventually be transmitted to a deterministic destination node, which will cache the ID-location pair and check for node clone detection, acting as an inspector. In addition, some intermediate nodes also behave as inspectors to improve resilience against the adversary in an efficient way.

E. Distributed Hash Table

Before diving into the detection protocol, Briefly introduce DHT techniques. In principle, a distributed hash table is a decentralized distributed system that provides a key-based lookup service similar to a hash table: (key, record) pairs are stored in the DHT, and any participating node can efficiently store and retrieve records associated with specific keys. By design, DHT distributes responsibility of maintaining the mapping from keys to records among nodes in an efficient and balanced way, which allows DHT to scale to extremely large networks and be suitable to serve as a facility of distributed node clone detection.

There are several different types of DHT proposals, such as CAN, Chord, and Pastry. Generally, CAN has least efficiency than others in terms of communication cost and scalability, and it is rarely employed in real systems. By contrast, Chord is widely used, and we choose Chord as a DHT implementation to demonstrate our protocol. However, our protocol can easily migrate to build upon Pastry and present similar se-curity and performance results.



Node	Key
N8	K114
	K1
N41	K15
	K31
N61	K52
N88	K79
N109	K97

N8's Finger Table

t	Key	Node
1	72	N88
2	40	N41
3	24	N41

Fig. 4: Chord network

Fig. 1.3. Chord network example, where the key space is 7 -bit $b=7$, seven records with different keys are stored in five nodes, and the successor table size $g=2$. For node $N8$, its direct predecessor is $N109$, and its two successors are $N41$ and $N61$.

By contrast, Chord is widely used, and we choose Chord as a DHT implementation to demonstrate our protocol. However, our protocol can easily migrate to build upon Pastry and present similar security and performance results. The technical core of Chord is to form a massive virtual ring in which every node is located at one point, owning a segment of the periphery. To achieve pseudo-randomness on output, a hash function is used to map an arbitrary input into a -bit space, which can be conceived as a ring

For node. As the kernel of efficient key-based routing, every node maintains a finger table of size t=O(log n) to facilitate a binary-tree search. Specifically, the finger table for a node with Chord coordinate contains information of nodes that are respectively responsible for holding the keys: (y+2b-1) mod 2b for I € [1, t]. If two nodes are within the ring segments distance, they are each other's predecessor and successor by the order of their coordinates, with respect to predefined g. In theory, a Chord node only needs to know its direct predecessor and finger table. To improve resilience against network churn and enhance routing efficiency, every node additionally maintains a successor table, g containing its successors.

F. PRELIMINARIES

1) PROTOCOL DETAILS

As a prerequisite, all nodes cooperatively build a Chord overlay network over the sensor network. Cloned node may not participate in this procedure, but it does not give them any advantage of avoiding detection. The construction of the overlay network is independent of node clone detection. As a result, nodes possess the information of their direct predecessor and successor in the Chord ring. In addition, each node caches information of its consecutive successors in its successors table. Many Chord systems utilize this kind of cache mechanism to reduce the communication cost and enhance systems robustness. More importantly in our protocol, the facility of the successors table contributes to the economical selection of inspectors.

2) DHT - BASED DETECTION PROTOCOL

The principle of our first distributed detection protocol is to make use of the DHT mechanism to form a decentralized caching and checking system that can effectively detect cloned nodes. Essentially, DHT enables sensor nodes to distributive construct an overlay network upon a physical sensor network and provides an efficient key-based routing within the overlay network. A message associated with a key will be transmitted through the overlay network to reach a destination node that is solely determined by the key; the source node does not need to specify or know which node a message's destination is the DHT key-based routing takes care of transportation details by the message's key.

$$M_{\mathrm{ACT}} = \mathrm{nonce},\ \mathrm{seed},\ \mathrm{time}, \{\mathrm{nonce} \parallel \mathrm{seed} \parallel \mathrm{time}\}_{K^{-1}_{\mathrm{initiator}}}.$$

More importantly, messages with a same key will be stored in one destination node. Those facts build the foundation for our first detection protocol. As a beginning of a round of DHT-based clone detection, the initiator broadcasts the action message including a random seed. Then, every observer constructs a claiming message for each neighbor node, which is referred to as an examinee of the observer and the message, and sends the message with probability independently.

IV. PROPOSED SCHEMES

The two novel, practical node clone detection protocols with different tradeoffs on network conditions and performance. The first proposal is based on a distributed hash table (DHT) by which a fully decentralized, key-based caching and checking system is constructed to catch cloned nodes. DHT-based protocol can detect node clone with high security level and holds strong resistance against adversary's attacks. The second protocol, named randomly directed exploration, is intended to provide highly efficient communication performance with adequate detection probability for dense sensor networks. In the protocol, initially nodes send claiming messages containing a neighbor-list along with a maximum hop limit to randomly selected neighbors; then, the subsequent message transmission is regulated by a probabilistic directed technique to approximately maintain a line property through the network as well as to incur sufficient randomness for better performance on communication and resilience against adversary. One detection round consists of three stages.

Stage 1: Initialization

To activate all nodes starting a new round of node clone detection, the initiator uses a broadcast authentication scheme to release an action message including a monotonously increasing nonce, a random round seed, and an action time. The nonce is intended to prevent adversaries

from launching a DoS attack by repeating broadcasting action messages. The action message is defined by

MACT = nonce, seed, time, {nonce || seed || time}k-1initiator

Stage 2: Claiming neighbors information

Upon receiving an action message, a node verifies if the message nonce is greater than last nonce and if the message signature is valid. If both pass, the node updates the nonce and stores the seed. At the designated action time, the node operates as an observer that generates a claiming message for each neighbor (examinee) and transmits the message through the overlay network with respect to the claiming probability . The claiming message by observer for examinee is constructed by where are locations of and , respectively. Nodes can start transmitting claiming messages at the same time, but then huge traffic may cause serious interference and degrade the network capacity. To relieve this problem, we may specify a sending period, during which nodes randomly pick up a transmission time for every claiming message.

Mα4β = idβ, Lβ, idα, Lα, {idβ || Lβ || idα || Lα || nonce }k-1α

Stage 3: Processing claiming messages

A claiming message will be forwarded to its destination node via several Chord intermediate nodes. Only those nodes in the overlay network layer (i.e., the source node, Chord intermediate nodes, and the destination node) need to process a message, whereas other nodes along the path simply route the message to temporary targets. Algorithm 1 for handling a message is the kernel of our DHT-based detection protocol. If the algorithm returns NIL, then the message has arrived at its destination. Otherwise, the message will be subsequently forwarded to the next node with the ID.

A. Buffer and Check Messages for Detection

1) Criteria of Determining Inspectors

The node acts as an inspector if one of the following conditions is satisfied.

Pseudo code for finding successor:

```
// ask node n to find the successor of id
n.find_successor(id)
        if (id ∈ (n, successor])
                return successor;
        else
                // forward the query around the
circle
                return
successor.find_successor(id);
```

Algorithm 1 :

DHT handle message(Mα4β) : handle a message in the DHT-based detection. Where is the current node"s succeeds key ((y + 2b-i) mod 2b), i ϵ [1, t], successor[j] is the next j th successor. j ϵ [1,g]

Output:

NIL if the message arrives at its destination:

Otherwise, it is the ID of the next node that receives the message in the chord overlay network

key <= H(seed || idβ)

if key ϵ (predecessor, y) then {has reached destination }

inspect (Mα4β) {act as an inspector, see Algorithm 2 }

return NIL

for i = 1 to g do

if key ϵ (y, successors[i]) then {destination is in the next Chord hop }

inspect(Mα4β) {act as an inspector, see algorithm }

return successors[i]

for j = 1 to t do {for normal DHT routing process }

if key ϵ [(y + 2b-i) mod 2b , y) then

return finger[j] return successors[g]

Algorithm 2:

Inspect (Mα4β) : Inspect a message to check for Clone detection in the DHT-based detection protocol

verify the signature of Mα4β

Pseudo code:

```
// ask node n to find the successor of id
n.find_successor(id)
        if (id ∈ (n, successor])
                return successor;
        else
                n' = closest_preceding_node(id);
                return n'.find_successor(id);
// search the local table for the highest predecessor of id
n.closest_preceding_node(id)
```

if idβ found in cache table then

if idβ has two distinct locations' {

clone, become a witness

}

broadcast the evidence

else

buffer Mα4β into cache table

```
        for i = m down to 1
                if (finger[i] ∈ (n, id))
                        return finger[i];
        return n;
```

1) This node is the destination node of the claiming message.

2) The destination node is one of the successors of the node.

In other words, the destination node will be reached in the next Chord hop. While the first criterion is intuitive, the second one is subtle and critical for the protocol performance. By Algorithm 1, roughly of all claiming messages related to a same examinee's ID will pass through one of the predecessors of the destination. Thus, those nodes are much more likely to be able to detect a clone than randomly selected inspectors. As a result, this criterion to decide inspectors can increase the average number of witnesses at a little extra memory cost. We will theoretically quantify those performance measurements later.

To examine a message for node clone detection, an inspector will invoke Algorithm 2, which compares the message with previous inspected messages that are buffered in the cache table. Naturally, all records in the cache table should have different examinee IDs, as implied in Algorithm 2. If detecting a clone, which means that there exist two messages Mα4β and Mα'4β' satisfying idβ=idβ' and Lβ ≠ Lβ , the witness node then broadcasts the evidence M evidence (Mα4β, Mα'4β') to notify the whole network. All integrity nodes verify the evidence message and stop communicating with the cloned nodes. To prevent cloned nodes from joining the network in the future, a evocation list of compromised nodes IDs may be maintained by nodes individually. It is worth noting that messages Mα4β and Mα'4β' are authenticated by observers α and α' , respectively.

2) Validity of Detection

The identity-based cryptographic system provides reliable identity authentication and message authentication for the DHT-based protocol. As a result, the adversary cannot falsify cloned nodes‟ IDs; neither can the modify messages signed by integrity nodes. Moreover, a cloned node cannot lie to its observers about its location since a forged location would be far deviated from the communication range of the observers, which suffices to alert observers. Therefore, the detection guidelines are robust provided observers are honest.

B. Broadcast Detection Evidence

1) Randomly Directed Exploration

The DHT-based detection protocol can be applied to general sensor networks, and its security level is remarkable, as cloned nodes will be caught by one deterministic witness plus several probabilistic witnesses. However, the message transmission over a Chord overlap network incurs considerable communication cost, which may not be desired for some sensor networks that are extremely sensitive to energy consumption.

The RDE protocol shares the major merit with broadcasting detection: Every node only needs to know and buffer a neighbor-list containing all neighbors IDs and locations. For both detection procedures, every node constructs a claiming message with signed version of its neighbor-list, and then tries to deliver the message to others which will compare with its own neighbor-list to detect clone. For a dense network, broadcasting will drive all neighbors of cloned nodes to find the attack, but in fact one witness that successfully catches the clone and then notifies the entire network would suffice

Algorithm 3:
RDE-process message(Mα) : An intermediate Node processes a message
Verify the signature of Mα
compare its own neighbor-list with the neighbor-list in Mα
if found clone then
broadcast the evidence;
tt1 <= tt1 – 1
If tt1 < 0 then
discard Mα
else
next node <= get next node(Mα) { see Algorithm 4}
if next node = NIL then
discard Mα
else
forward Mα to next node

2) Protocol Description

One round of clone detection is still activated by the initiator. Subsequently, at the designated action time, each node creates its own neighbor-list including the neighbors IDs and locations, which constitutes the sole storage consumption of the protocol. The claiming message by node is constructed By

Mα=tt1,idα,Lα,Neighor Iistα, {idα,||Lα||Neighbor Listα,||nonce}kα-1

where tt1 is time to live (a.k.a. message maximum hop). Since tt1 will be altered by intermediate nodes during transmission, it should not be authenticated. The observer will deliver the claiming message times. In each time, the node transmits it to a random neighbor as indicated in, note that can be a real number, and accordingly an observer transmits its claiming message at least [ɤ], up to[ɤ], and on average times. When an intermediate node β receives a claiming message Mα, it launches RDE process message (Mα), which is described by pseudo code in Algorithm 3, to process the message. During the processing, node β, as an inspector, compares its own neighbor-list to the neighbor list in the message, checking if there is a clone.

Performance on Different Network Sizes:

To develop the first simulation, which not only measures the randomly directed exploration protocol's performance on varying network sizes, but also verifies if our protocol design really fulfills the intention? In order to do that, we mix the three routing mechanisms as the following three groups, only the mechanism of deterministic directed transmission is used. Consequently, all claiming messages, if not dropped by malicious nodes, go exactly tt1 hops. Next, group II adds network border determination into group I. As the network shape is a regular square. Finally, group III further adds probabilistic directed transmission into group II, and indeed constitutes Algorithm Each node n' maintains a routing table with up to m entries (which is in fact the number of bits in identifiers), called finger table.

The ith entry in the table at node n contains the identity of the first node s that succeeds n by at least 2i-1 on the identifier circle.

s = successor($n+2^{i-1}$).
3.4.3 Node Joins – join()
s is called the i^{th} finger of node n, denoted by n.finger(i)
// called periodically. refreshes finger table entries.
n.fix_fingers()
next = next + 1 ;

```
    if (next > m)
            next = 1 ;
    finger[next] = find_successor(n + 2^(next-1));
// checks whether predecessor has failed.
n.check_predecessor()
    if (predecessor has failed)

    predecessor = nil;
```

V. CONCLUSION

Sensor nodes lack tamper resistant hardware and are subject to the node clone attack. This project deals two distributed detection protocols. One is based on a distributed hash table, which forms a Chord overlay network and provides the key-based routing, caching, and checking facilities for clone detection, and the other uses 1probabilistic directed technique to achieve efficient communication overhead for satisfactory detection probability. While the DHT-based protocol provides high security level for all kinds of sensor networks by one deterministic witness and additional memory efficient, probabilistic witnesses, the randomly directed exploration presents outstanding communication performance and minimal storage consumption for dense sensor networks.

VI. FUTURE ENHANCEMENT

Providing reliable security mechanisms for large-scale sensor networks of low-cost nodes is an interesting and challenging research area. While many issues have been addressed successfully, other problems still remain open and need further study. On the other hand, many techniques behind our work in this thesis may be extended to other new rising application areas with added efforts

REFERENCES

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

[2]. H. Balakrishnan, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica, "Looking up data in P2P systems,"Commun. ACM, vol. 46, no. 2, pp. 43–48, 2003.

[3]. Y.Zhang, W.Liu, W.Lou, and Y.Fang, "Location based compromise tolerant security mechanisms for wireless sensor networks," IEEE J. Sel. Areas Commun., vol. 24, no. 2, pp. 247–260, Feb. 2006.

[4]. S.Zhu, S.Setia, and S. Jajodia, "LEAP: Efficient security mechanisms for large scale distributed sensor networks," in Proc. 10th ACM CCS , Washington, DC, 2003, pp. 62–72.

[5]. R. Anderson, H. Chan, and A. Perrig, "Key infection: Smart trust for smart dust," in Proc. 12th IEEE ICNP, 2004, pp. 206–215.

[6]. 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," inProc.8th ACM Mobi Hoc, Montreal, QC, Canada, 2007, pp. 80–89.

[7]. B. Zhu, V. G. K. Addada, S. Setia, S. Jajodia, and S. Roy, "Efficient distributed detection of node replication attacks in sensor networks," in Proc. 23rd ACSAC, 2007, pp. 257–267.

[8]. H. Choi, S. Zhu, and T. F. La Porta, "SET: Detecting node clones in sensor networks," in Proc. 3rd Secure Comm, 2007, pp. 341–350. 50

[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 Trans. Syst.s, Man, Cybern. C, Appl. Rev., vol. 37, no. 6, pp. 1246–1258, Nov. 2007.

[10]. L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in Proc. 9th ACM Conf. Comput. Commun. Security, Washington, DC, 2002, pp. 41–47.

[11]. A. Shamir, "Identity-based cryptosystems and signature schemes," in Proc. CRYPTO, 1984, LNCS 196, pp. 47–53.

[12]. R. Poovendran, C. Wang, and S. Roy, Secure Localization and Time Synchronization for Wireless Sensor and Ad Hoc Networks. New York: Springer Verlag, 2007.

[13]. I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," IEEE Commun. Mag., vol. 40, no. 8, pp. 102–114, Aug. 2002.