Distributed Clone Detection in Mobile Sensor Networks

Abu Sayed Chowdhury[1] and Akshaye Dhawan[2]

[1] *School of Electrical Engineering and Computer Science, Washington State University, P.O. Box 642752, Pullman, WA, USA*

[2] *Ursinus College,Department of Mathematics and Computer Science, 601 E. Main Street, Collegeville, PA, USA*

achowdhu@eecs.wsu.edu, adhawan@ursinus.edu

Keywords: Security, Mobile Wireless Sensor Networks, Clone attacks, Distributed Detection Protocol

Abstract: In this paper we present a clone detection technique for mobile wireless sensor networks. Since sensor nodes can be compromised and cloned by an adversary, it is desirable to have a distributed clone detection algorithm wherein the nodes in the network themselves identify cloned nodes. Most approaches in the literature have a deterministic set of witness nodes that check for the presence of cloned nodes for a given ID based by receiving claim messages that show two or more nodes with the same ID in different locations. We present a new approach to distributed clone detection called Distributed Detection using Prefix Matching (DDPM). Our approach is designed to work in a mobile sensor network and is based on using a random number and a hash function of the claiming nodes ID to generate a key. We then match the prefix of this key to sensor node IDs in the network to determine a set of inspector nodes that will detect clones for a given round. We demonstrate a 7-10% improvement in the number of witness nodes as a measure of security while also reducing the communication overhead.

1 INTRODUCTION

In recent years, mobile sensor networks are being utilized in several applications including surveillance, habitat monitoring, object tracking, and environmental information gathering (Akyildiz et al., 2002; Bonaci et al., 2010; Zhou et al., 2006). Since sensor nodes are often deployed in remote and harsh areas, the sensors are vulnerable to capture and compromise. Once captured both the data and the code on a sensor node is potentially available to an adversary. Attacks can be launched by cloning a captured sensor node and using the cloned nodes to launch a variety of insider attacks, injecting false data and conducting incorrect aggregation. Such attacks have the potential to severely impact the networks functioning.

Although clone attack detection in static networks has been explored in the literature, limited work has been done to tackle this security threat in a mobile sensor network. It is not straightforward to apply the clone detection techniques developed for static networks to a mobile environment. Some centralized approaches have been introduced in the literature to identify clones in mobile sensor networks; however, they suffer from having a single point of failure and excessive communication costs. Furthermore, centralized methods are not appropriate in mobile sensor network when nodes are scattered in different regions of the network (Yi et al., 2008). Hence, distributed detection algorithms are crucial to detect cloned nodes in a mobile network.

In this paper, we present a distributed clone detection strategy called the Distributed Detection Prefix Matching (DDPM) protocol. The key idea of this technique is to implement a decentralized key based verification protocol to identify clone nodes. We also conduct a simulation study comparing DDPM to existing protocols for both communication overhead and the level of security provided. We organize the remainder of this paper as follows. We categorize and illustrate related work in Section 2. Details of DDPM protocol are described in Section 3. Simulation results and performance comparison of DDPM with XED (Yu et al., 2013) are shown in Section 4. Finally, we conclude the paper and discuss future work in Section 5.

2 RELATED WORK

In this section, we present a brief survey of the current literature on clone detection. We classify clone detection approaches into two categories. The first category includes centralized algorithms where a central base station identifies a cloned node by receiving reports from the nodes of the sensor network. The second category includes distributed algorithms where information is processed by the nodes in the network cooperatively in order to identify clones in a distributed manner.

2.1 Centralized Detection

One of the first centralized algorithms for clone detection in the literature was presented in (Eschenauer and Gligor, 2002). In this paper the authors present a detection scheme where every sensor node in the network sends a list of its neighboring nodes to the base station (BS). The BS considers two nodes as clone nodes if it receives two lists that contain the same node ID with inconsistent node positions. The major drawbacks of this solution is that it suffers from a single point of failure (the BS) and also has an excessive communication overhead due to many messages exchanged between nodes and the BS. Additionally, the nodes closer to the base station are responsible to forward messages to the BS, thereby consuming more energy than other nodes and resulting in a shorter lifetime for these nodes.

The authors in (Brooks et al., 2007) introduce a detection algorithm based on a random key distribution approach. Here, every sensor node is given a set of randomly chosen symmetric keys and those keys are selected from a larger pool of keys. A counting bloom filter is used by every node for clone detection and the base station knows the filter of every sensor node. In order to detect clones, the base station examines all keys used in the network and if each of these exceed a certain threshold then that key is considered as cloned. Finally, a revocation procedure is called to detect the corresponding cloned nodes.

The SET protocol (Choi et al., 2007) models the network as a set of non-overlapping regions or clusters. The base station broadcasts a random value which is used to construct clusters or subsets and select cluster heads. Cluster heads are referred to as subset leaders (SLs) and clusters are formed using an exclusive subset maximal independent set (ESMIS) algorithm. One or more trees are constructed with SLs and a bottom-up aggregation is used to aggregate all nodes. A node is said to be cloned if the corresponding ID is found in two distinct subsets. SET suffers from a high communication overhead due to a complex detection scheme.

All of the protocols described above are used in static sensor networks but they are not suitable for mobile nodes. (Ho et al., 2009) was the first work to look at clone detection in mobile networks by presenting a Sequential Probability Ratio Test (SPRT) based clone detection algorithm. In this scheme, all nodes report to the Base Station (BS) with a list of encountered nodes every time they change location. The BS checks for a cloned node by seeing if it is present in two different locations with a velocity surpassing a certain threshold. This approach could be error-prone due to mistakes in node speed measurements and SPRT is used to avoid this. However, this approach still involves a single point failure as well as rapid energy depletion of nodes near the BS.

2.2 Distributed Detection

Line Selected Multicase (LSM) (Parno et al., 2005) is the first globally aware distributed algorithm for clone detection. In LSM, the location of a node is forwarded with a certain probability to some randomly chosen destination nodes. Some intermediate nodes form a claim message path to forward claim messages from the source to the destination node. All forwarding nodes examine the signature of the claim, save it and verify its coherence with other position claims. A clone is said to be detected when a node in the intersection of two message paths discovers two different nodes (with the same ID) with two distinct claims. LSM suffers from a high memory overhead and a high communication overhead due to the storing of many messages and the node-to-node message passing. Also, it assumes that all nodes are aware of each others location which may restrict its applicability.

In the RED (Randomized, Efficient and Distributed) (Conti et al., 2011) protocol, the base station broadcasts a pseudo-random number to every sensor node periodically. This protocol consists of two steps for detecting clone nodes. In the first step, a random value is shared between nodes and then every node constructs a digital signature and locally broadcasts its claim-ID and geographical position to a node set known as the witness nodes. Witness nodes are responsible for clone detection. In the second step, these witness nodes, verify the received signature and check each claim message for freshness. This enables clone detection and also prevents stale messages from being replayed. It is worth noting that this approach is deterministic i.e., the nodes that will act as inspectors to identify suspicious activities are fixed and that this strategy is also used for static wireless sensor networks.

Zhu *et al.* (Zhu et al., 2007) presented two methods Single Deterministic Cell (SDC) and Parallel Multiple Probabilistic Cell (P-MPC) where the network is split into cells and every cell is used to map nodes. When a node claims, it is routed to the mapped cell and the claiming message is broadcast to the mapped cell. The nodes within the mapped cell then acts as witnesses for the claiming node. The difference between P-MPC with SDC is that multiple cells can be used to map each node ID; however, cell mapping is still deterministic since the location claim is mapped and sent to multiple deterministic cells with varying probabilities.

The primary version (Yu et al., 2008) of (Yu et al., 2009) describes the first distributed clone detection technique for a mobile sensor network. This version applies a challenge-and-response technique but its detection efficiency is susceptible to colluding clones and (Yu et al., 2009) provides an approach to detect the colluding clones. But the drawback of this approach is that the storage space requirement grows linearly with the network size. Another limitation is the scalability of this approach as the network size grows.

Yu *et al.* (Yu et al., 2013) proposed two algorithms eXtremely Efficient Detection (XED) and Efficient Distributed Detection (EDD) for mobile sensor networks to resist clone attacks in a localized fashion. Every node sends its node ID to the base station using its one-hop neighbors. The base station then checks all the IDs it receives for clone detection. However, they require a double signature scheme which incurs additional communication costs.

3 DISTRIBUTED DETECTION USING PREFIX MATCHING (DDPM)

3.1 Preliminaries

We consider a set of mobile nodes with node IDs $\{1,...,n\}$ in a homogeneous mobile sensor network. We assume symmetric communication. Time for all of the sensor nodes is divided into intervals of equal length and it is not necessary for time to be synchronized across nodes. The sensor nodes use a random way point (RWP) model (Johnson and Maltz, 1996) to move wherein a node moves towards a destination position with a speed $\bar{v}$ chosen arbitrarily from the interval $[\bar{v}_{min}, \bar{v}_{max}]$. A node does not change its position for a random amount of time when it reaches the destination point. When a node starts a clone detection round it only moves after this round ends. We assume that every node is aware of its own geographic position and some neighbors are in close proximity to a node on every move. We also assume that the nodes in the mobile sensor network construct an overlay network that can be used for routing. We utilize an identity-based public key system (Liu and Ning, 2008; Malan et al., 2008) to generate and verify signatures. It is not mandatory to have a powerful base station (BS) but an initiator should initiate the distributed detection procedure so that an intruder cannot conduct a denial-of-service (DoS) threat by insisting that beginning the clone detection procedure would result in depletion of the cloned nodes energy.

3.2 Threat Model

We consider an adversary model where an intruder can capture and compromise sensor nodes that are placed in hostile environment. However, the adversary can compromise only a few nodes to create clones and will deploy these clones in the network intelligently. We must have a minimum of at least one legitimate node for each cloned node in order for the proposed scheme to work.

The adversary will always want to hide cloned nodes from being detected and the detection procedure may be interrupted by the intruder as follows. Cloned nodes do not wish to participate in the detection procedure or cloned nodes try to fake, drop or manipulate claim messages that are sent as part of the detection procedure. Furthermore, it will take some time for an adversary to capture and compromise some nodes. Nodes excluding the compromised nodes are considered as *legitimate nodes*. Sometimes, an adversary can mislead the clone detection protocol to consider legitimate nodes as cloned nodes. We refer to this attack as a *framing attack*.

3.3 Performance Metrics

The metrics that are used to assess DDPM's performance are as follows:

- Security Level: Every clone detection protocol should detect node replication attacks with a high probability. If a detection approach is deterministic in a manner that cloned nodes are always detected by witness nodes and every node can act as a witness with equal probability, then the number of witness nodes can be used to measure the security performance since a large number of witness nodes can be used to resist a replication attack.
- Communication Overhead: We should minimize the communication cost to get better performance in a mobile sensor network because sending more messages consumes more energy in the sensor

nodes. For the purpose of this paper, the communication cost is represented as the average number of bits transmitted per node per move.

3.4 Reporter-Claimer-Witness model

We utilize the Reporter-Claimer-Witness model (Khan et al., 2015) for distributed detection. In this model, a *claimer* node broadcasts its location claim and ID to its neighbors. One or more of these neighbors serve as *reporters* and map the claim id to *witness* nodes. These *witness* nodes are responsible for clone detection. When they receive conflicting claims from cloned nodes, they start a revocation process to let other nodes know about the presence of cloned nodes.

A key problem that we sought to solve was that of avoiding having a deterministic set of witness nodes to verify a claim. If the set of witness nodes is deterministic this creates a significant threat since an adversary can capture and clone these nodes. In order to solve this problem, we created a prefix matching scheme that selects the set of witness nodes for each claim message differently for every detection round.

Every node is assigned a unique 128-bit node ID. We use a cryptographic hash function of the node IP address or public key to produce the node ID. For every claim message produced by a claiming node, a key is produced using a 128 bit hash by its neighboring reporter nodes. This key uses the random number that was sent by the initiator for clone detection as well as information from the claiming nodes claim message. The key idea in prefix matching is to have the reporter nodes compute a hash for the claim message and then search for the node whose node ID is mathematically nearest to the hash-value. This node will serve as the witness and check for claims. In prefix matching, we pick multiple witness nodes by picking nodes that match the generated key for each claim message in increasing lengths. This process is explained in more detail using an example in the next subsection.

3.5 Details of DDPM

When a new detection round begins, the initiator (usually the base station) broadcasts an action message containing a random number, *rand*.

In DDPM, there are three steps in every round of clone detection. These are Initialization, Claiming Neighbors and Processing Claims. We now present each of these in more detail.

3.5.1 Initialization

In every new round of clone identification, the initiator broadcasts an initialization message to start a detection round. The initialization message is given by:

$$ini\_message := < rand, time, signed(k^{Priv}_{Initiator}) >$$

In the initialization message *rand* is the random number generated by the initiator, *time* is the time stamp when the initiator generated this message and the message is signed with $k^{Priv}_{Initiator}$ which is the private key of the initiator. This message is then broadcast over the network.

3.5.2 Claiming Neighbors

When a sensor node receives the initialization message, it verifies if the message signature is valid or not. If valid, it extracts the random number *rand* from the message. Each node then produces a claim message with its location and id and sends this to its neighboring nodes who will serve as reporters. The reporters will take the claim message and send this to a set of witness nodes who will verify that the claiming node is who it claims to be. By doing this, each sensor node acts as a reporter node for its neighbors. The claim message is sent through the overlay network with a probability ρ_c. ρ_c can be seen as a means to control the tradeoff between network traffic and security. When $\rho_c = 1$, every reporter node will forward the claim message for its neighbor. For other values of ρ_c, the value determines the probability of a reporter node forwarding the claim message. For example, when $\rho_c = 0.5$, there is a probability of 0.5 that a reporter will forward the claim message, thereby reducing the number of nodes participating in the clone detection process.

Now let us look into the details of a claim message. Suppose the reporter is node A and it is examining the claimer node B. Node A will create the following claim message :

$$claim_{BA} := < Id_B, Loc_B, Id_A, Loc_A, signed(K^{Priv}_A) >$$

In the above claim message Id_B, Loc_B, Id_A, Loc_A are the ID of node B, the location of node B, the ID of node A, the location of node A respectively and the message is signed by the private key of node A given by K^{Priv}_A. The reporter node A then generates a key for the claiming message using the rand value sent by the initiator as follows:
$key := \ H(rand, Id_B)$

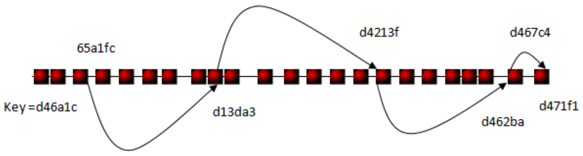

Figure 1: Illustrating verification of claiming message.

This key along with the claim message is used in the next step. It is worth mentioning that for two different cloned nodes having the same ID but existing at different locations, the claim message produced in this step will vary but the key will be identical since it is based on the random number used in a given round of clone detection and the ID of the node.

3.5.3 Processing Claims

Next we present our distributed detection algorithm that deals with the claim messages generated in the previous step. Every node in the mobile wireless sensor network contains a table with routing information, a table with information about its neighboring nodes and a leaf set. The routing table and leaf set are computed for a given claim message key. In the routing table, the entries at a given row i point to the nodes that share the first i characters of their ID's with the message key (hence the name prefix matching). Also, the leaf set contains an equal number of closest nodes with a larger ID and closest nodes with a smaller ID than this node.

Our distributed detection algorithm is presented in Algorithm 1 and it uses and associated clone detection routing presented in Algorithm 2. R^{d}_{length} indicates a node in the routing table R at row $length$ and column d. d is the digit of the key at position $length$. Each node upon receiving the initialization message from the initiator generates a claim message and its associated key as explained in the previous subsection. To avoid using a deterministic set of witness nodes to verify this claim, our algorithm uses the key associated with the claim message to pick a set of nodes that will serve as witnesses based on matching the prefix of their ID's to that of the claim message key. Since the key is used to pick the set of witness nodes and the cloned nodes have the same keys, the witness nodes will be able to detect the presence of cloned nodes. Furthermore, since a new set of witness nodes is picked each round based on the random number used by the initiator, it becomes difficult for an adversary to guess what nodes will serve as witness nodes.

An example: Let us now look at an example to better understand how the process of using prefix matching to forward claim messages works. Fig. 1 depicts the process of forwarding claim messages. Suppose that the source node is sensor A and has an ID of $65a1fc$ and it is generating a claim message for a neighbor B. Also suppose that the key for this claim message generated for B is $d46a1c$. Node A will forward this message to the node which has one more matching character with the key, i.e., to $d13da3$ (Since A has 0 digits matching with the message key, this node $d13da3$ is incrementing the matching length to 1). Subsequently the message will be forwarded to the nodes $d4213f$, $d462ba$ and $d467c4$ respectively since each of these nodes is either improving the matched length between its id and the message key by one or is closest to the message key (in the case of $d467c4$). All of these nodes are responsible for clone detection. Note that since this key was generated using the random number $rand$ sent by the initiator, each time clone detection is performed, a different set of nodes will be selected to serve as witness nodes and verify that claim to detect clones.

Algorithm 1 Claim management algorithm: *claim_manage*

```
for each neighbor $B$ of $Id_A$ do
    $claim_{BA}$:= $< Id_B, Loc_B, Id_A, Loc_A, signed(K_A^{Priv}) >$;
    key:= $H(rand, Id_B)$;
    if key destination is within its leaf set then
        forward to the closest node Id of its leaf set;
        $detect\_clone$(memory, $claim_{BA}$);
    else
        length:= prefix($key, Id_A$);
        if $R^{d}_{length}$ exists then
            forward to the node of $R^{d}_{length}$;
            $detect\_clone$(memory, $claim_{BA}$);
        else
            forward to a node whose $prefix(key, id)$ $\geq length$ and is numerically closer to the key;
            $detect\_clone$(memory, $claim_{BA}$);
        end if
    end if
end for
```

Algorithm 2 Clone detection algorithm: *detect_clone*

```
check the signature of $claim_{BA}$;
if $Id_B$ has found in the memory then
    if $Id_B$ has found in two distinct $Loc$ then
        broadcast revocation procedure;
    else
        store $claim_{BA}$ in memory;
    end if
end if
```

3.6 Security Issues

3.6.1 Detection Validation

The identity-based cryptographic approach used in the above algorithms is very useful when the authentication of a trusted identity and claim message are crucial. Also, this makes it difficult for the adversary to falsify its node ID or to alter the messages signed by legitimate nodes. Moreover, it will be tough for clones to fake their location to the inspector as a falsely advertised position is far enough deviated from the transmission range of the witnesses that is sufficient to alert the inspectors.

3.6.2 Witness Protection

When the base station or initiator broadcasts a random number, the adversary may try to compromise witness nodes to defeat detection. Although the intruder may try to use the random number to compute who the witness nodes are and then compromise them to fail detection, this is unlikely. Since nodes are randomly deployed in the network, it is not possible for the adversary to capture all of the witness nodes. Additionally, for each detection round the witness nodes are changed so that the intruder cannot postpone clone detection by capturing some witness nodes.

4 SIMULATION RESULTS AND ANALYSIS

In this section, we evaluate the performance of our Distributed Detection using Prefix Matching (DDPM) strategy via computer simulation and compare it with eXtremely Efficient Detection (XED) presented in (Yu et al., 2013).

4.1 Simulation Setup

We implement DDPM using the discrete event network simulator OMNeT++ (OpenSim, 2016). We utilize the random graph model to generate our simulation network. In the simulation, we adjust the sensor node communication dynamically to maintain an approximate node degree d. We keep the approximate node degree $d = 15$ for our simulations. The claiming probability, ρ_c is used as 1.0 for pro-security. This implies that every claim message will be forwarded. We set the network size $N = 1000$ nodes. To get precise results, we consider 8 different network instances, and for each simulation run of these instances, we perform 10 rounds of detection. A random number *rand* is generated for every round and we randomly select the number of cloned nodes between $2 - 100$ by setting their id the same as that for another node.

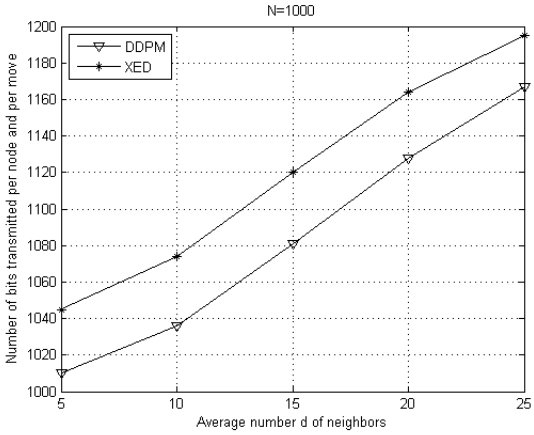

Figure 2: Communication overhead comparison among DDPM and XED (N = 1000).

4.2 Performance Evaluation

To evaluate DDPM against XED in mobile sensor networks, we consider two performance metrics. The first metric is the average number of witness nodes and this is used to evaluate the security level. The second metric is the communication overhead. Both were explained in Section 3. We vary the number of cloned nodes randomly between 2 to 100.

Fig. 2 depicts the communication overhead of DDPM and XED. Nodes change positions according to random waypoint model. XED incurs a higher communication costs because nodes need to exchange a random number using hash function. In XED when a node encounters another node, it needs to compute the hash function again to verify the legitimacy of the encountered node. On the other hand, DDPM requires each node to communicate with some intermediate nodes to check for clones and the hash function is applied once during the transmission. We see a 4-5% reduction in the communication overhead for DDPM when compared to XED.

Fig. 3 depicts the simulation results for the security level metric. The average number of witnesses in DDPM is pretty high even if 15% of the sensors maliciously reject claim messages. The performance of DDPM is compared with extremely efficient detection (XED). The simulation results indicate that the average number of witness for the DDPM protocol is 7%-10% higher than that for XED. The reason for this is that in XED, only neighbors are responsible to check whether any node has been cloned or not whereas in

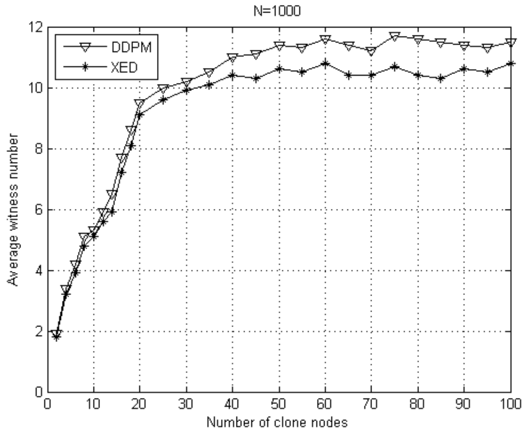

Figure 3: Security level comparison among DDPM and XED (N = 1000).

DDPM, the neighbors, the intermediate nodes based on prefix and the destination node all serve as witness. Furthermore this set of nodes varies for each round of detection and is not predetermined. This exhibits the superiority of the DDPM.

5 CONCLUSION

In this paper, we introduced a distributed approach for clone detection in mobile sensor networks. We aimed to solve the problem of avoiding picking a deterministic set of nodes that existing approaches suffer from. Our initial simulations show that DDPM can outperform existing state-of-art approached like XED with by providing a higher security level. DDPM also reduces the communication overhead. DDPM is ID-oblivious as well as area oblivious. For our future work, we will measure the performance by incorporating several independent DDPM systems with different random numbers and vary the network density to check performance in the case of sparse mobile sensor network.

REFERENCES

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cayirci, E. (2002). A survey on sensor networks. *IEEE communications magazine*, 40(8):102–114.

Bonaci, T., Bushnell, L., and Poovendran, R. (2010). Node capture attacks in wireless sensor networks: a system theoretic approach. In *49th IEEE Conference on Decision and Control (CDC)*, pages 6765–6772. IEEE.

Brooks, R., Govindaraju, P., Pirretti, M., Vijaykrishnan, N., and Kandemir, M. T. (2007). On the detection of clones in sensor networks using random key predistribution. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(6):1246–1258.

Choi, H., Zhu, S., and La Porta, T. F. (2007). Set: Detecting node clones in sensor networks. In *Security and Privacy in Communications Networks and the Workshops, 2007. SecureComm 2007. Third International Conference on*, pages 341–350. IEEE.

Conti, M., Di Pietro, R., Mancini, L., and Mei, A. (2011). Distributed detection of clone attacks in wireless sensor networks. *IEEE Transactions on Dependable and secure computing*, 8(5):685–698.

Eschenauer, L. and Gligor, V. D. (2002). A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM conference on Computer and communications security*, pages 41–47. ACM.

Ho, J. W., Wright, M., and Das, S. K. (2009). Fast detection of replica node attacks in mobile sensor networks using sequential analysis. In *INFOCOM 2009, IEEE*, pages 1773–1781. IEEE.

Johnson, D. B. and Maltz, D. A. (1996). Dynamic source routing in ad hoc wireless networks. In *Mobile computing*, pages 153–181. Springer.

Khan, W. Z., Aalsalem, M. Y., and Saad, N. (2015). Distributed clone detection in static wireless sensor networks: Random walk with network division. *PLoS One*, 10(5).

Liu, A. and Ning, P. (2008). Tinyecc: A configurable library for elliptic curve cryptography in wireless sensor networks. In *Information Processing in Sensor Networks, 2008. IPSN'08. International Conference on*, pages 245–256. IEEE.

Malan, D. J., Welsh, M., and Smith, M. D. (2008). Implementing public-key infrastructure for sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 4(4):22.

OpenSim (2016). Omnet++.

Parno, B., Perrig, A., and Gligor, V. (2005). Distributed detection of node replication attacks in sensor networks. In *2005 IEEE Symposium on Security and Privacy (S&P'05)*, pages 49–63. IEEE.

Yi, J., Koo, J., and Cha, H. (2008). A localization technique for mobile sensor networks using archived anchor information. In *2008 5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, pages 64–72. IEEE.

Yu, C.-M., Lu, C.-S., and Kuo, S.-Y. (2008). Mobile sensor network resilient against node replication attacks. In *2008 5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, pages 597–599. IEEE.

Yu, C.-M., Lu, C.-S., and Kuo, S.-Y. (2009). Efficient and distributed detection of node replication attacks in mobile sensor networks. In *Vehicular Technology Conference Fall (VTC 2009-Fall), 2009 IEEE 70th*, pages 1–5. IEEE.

Yu, C.-M., Tsou, Y.-T., Lu, C.-S., and Kuo, S.-Y. (2013). Localized algorithms for detection of node replication attacks in mobile sensor networks. *IEEE transactions on information forensics and security*, 8(5):754–768.

Zhou, L., Ni, J., and Ravishankar, C. V. (2006). Supporting secure communication and data collection in mobile sensor networks. In *INFOCOM*.

Zhu, B., Addada, V. G. K., Setia, S., Jajodia, S., and Roy, S. (2007). Efficient distributed detection of node replication attacks in sensor networks. In *Computer Security Applications Conference, 2007. ACSAC 2007. Twenty-Third Annual*, pages 257–267. IEEE.