

Fast Detection of Replica Node Attacks in Mobile Sensor Networks Using Sequential Analysis

Abstract—Due to the unattended nature of wireless sensor networks, an adversary can capture and compromise sensor nodes, generate replicas of those nodes, and mount a variety of attacks with the replicas he injects into the network. These attacks are dangerous because they allow the attacker to leverage the compromise of a few nodes to exert control over much of the network. Several replica node detection schemes in the literature have been proposed to defend against these attacks in static sensor networks. These approaches rely on fixed sensor locations and hence do not work in mobile sensor networks, where sensors are expected to move. In this work, we propose a fast and effective mobile replica node detection scheme using the Sequential Probability Ratio Test. To the best of our knowledge, this is the first work to tackle the problem of replica node attacks in mobile sensor networks. We show analytically and through simulation experiments that our schemes achieve effective and robust replica detection capability with reasonable overheads.

I. INTRODUCTION

Advances in robotics have made it possible to develop a variety of new architectures for autonomous wireless networks of sensors. Mobile nodes, essentially small robots with sensing, wireless communications, and movement capabilities, are useful for tasks such as static sensor deployment, adaptive sampling, network repair, and event detection [9]. These advanced sensor network architectures could be used for a variety of applications including intruder detection, border monitoring, and military patrols. In these kinds of hostile or potentially hostile environments, the security of unattended mobile nodes is critical. The attacker may be able to capture and compromise mobile nodes, and then he can use them to inject fake data, disrupt network operations, and eavesdrop on network communications.

In this scenario, a particularly dangerous attack is the *replica attack*, in which the adversary takes the secret keying materials from a compromised node, generates a large number of attacker-controlled replicas that share the node's keying materials and ID, and then spreads these replicas throughout the network. With a single captured node, the adversary can create as many replica nodes as he has the hardware to generate. Note that replica nodes need not be identical robots; a group of static nodes can mimic the movement of a robot and other mobile nodes or even humans with handheld devices could be used. The only requirement is that they have the software and keying material to communicate in the network, all of which can be obtained from the captured node.

The time and effort needed to inject these replica nodes into the network should be much less than the effort to capture and compromise the equivalent number of original nodes. The replica nodes are controlled by the adversary, but have

keying materials that allow them to seem like authorized participants in the network. Protocols for secure sensor network communication would allow replica nodes to create pairwise shared keys with other nodes and the base station, enabling the nodes to encrypt, decrypt, and authenticate all of their communications as if they were the original captured node.

The adversary can then leverage this insider position in many ways. For example, he can simply monitor a significant fraction of the network traffic that would pass through these nodes. Alternately, he could jam legitimate signals from benign nodes or inject falsified data to corrupt the sensors' monitoring operation. A more aggressive attacker could undermine common network protocols, including cluster formation, localization, and data aggregation, thereby causing continual disruption to the network's operations. Through these methods, an adversary with a large number of replica nodes can easily defeat the main mission of the deployed network.

A straightforward solution to stop replica attacks is to prevent the adversary from extracting secret key materials from mobile nodes by equipping them with tamper-resistant hardware. We might expect such measures to be implemented in mobile nodes with security-critical missions. However, although tamper-resistant hardware can make it significantly harder and more time-consuming to extract keying materials from captured nodes, it may still be possible to bypass tamper resistance for a small number of nodes given enough time and attacker expertise. Since the adversary can generate many replicas from a single captured node, this means that replica attacks are even more dangerous relative to attempting to compromise many nodes. We thus believe that it is very important to develop software-based countermeasures to defend mobile sensor networks against replica attacks.

Several software-based replica node detection schemes have been proposed for static sensor networks [6], [8], [20], [28]. The primary method used by these schemes is to have nodes report *location claims* that identify their position and attempt to detect conflicting reports that signal one node in multiple locations. However, since this approach requires fixed node locations, it cannot be used when nodes are expected to move. Thus, our challenge is to design an effective, fast, and robust replica detection scheme specifically for mobile sensor networks.

In this paper, we propose a novel mobile replica detection scheme based on the *Sequential Probability Ratio Test* (SPRT). We use the intuition that an uncompromised mobile node should move at speeds in excess of the system-configured maximum speed. If we observe that a mobile node's speed is

over the maximum speed, it is then highly likely that at least two nodes with the same identity are present in the network. Specifically, we perform the SPRT on every mobile node using a null hypothesis that the mobile node hasn't been replicated and an alternate hypothesis that it has. We show in this work how to configure upper and lower limits that allow the system to choose the right hypothesis for fast, accurate detection.

To the best of our knowledge, this work is the first attempt to tackle the replica detection problem in mobile sensor networks. We validate the effectiveness and efficiency of our scheme through analysis and simulation experiments. In particular, simulation results show that our scheme very quickly detects mobile replicas with zero false positive and negatives. This is mainly because the SPRT is proven to be the best mechanism in terms of the number of observations to reach a decision among all sequential and non-sequential decision processes.

The rest of paper is organized as follows. Section II describes the problem statement, assumptions, and adversary model for our schemes. In Section III, we will present our mobile replica detection scheme using the SPRT and analyze its security and performance. Section IV presents the results of simulations we conducted to evaluate the scheme. Section V presents the related work. Finally, Section VI concludes the paper and discusses some possible future work.

II. PRELIMINARIES

In this section, we first present the problem statement and underlying assumptions for our proposed scheme and then describe the attacker models and design goals.

1) *Problem Statement*: In this work, we tackle the problem of mobile replica node attacks. We define a mobile replica node u' as a node having the same ID and secret keying materials as a mobile node u . An adversary creates replica node u' as follows: He first compromises node u and extracts all secret keying materials from it. Then he prepares a new node u' , sets the ID of u' the same as u , and loads u 's secret keying materials into u' .

2) *Network Models*: We consider a two-dimensional *mobile* sensor network where sensor nodes freely roam throughout the network. We assume that every mobile sensor node's movement is physically limited by the system-configured maximum speed. We also assume that all direct communication links between sensor nodes are bidirectional. This communication model is common in the current generation of sensor networks.

Several mobility models [2], [3], [4] have been proposed to evaluate the performance of protocols in mobile ad hoc networks (MANET). Among these models, the Random Waypoint Mobility (RWM) model is widely used in the literature. However, according to [27], when the RWM model is used in simulation, it takes some time for the probability distribution of the movement of nodes to converge to a steady-state distribution after the start of simulation. Furthermore, the convergence time is changed in accordance with the parameters of the mobility model and the performance of the network varies with the convergence time. Thus, it is hard to find a steady-state distribution in the RWM model. To resolve

this problem, Le Boudec et al. [2] propose the Random Trip Mobility (RTM) model as a generic framework for finding the steady-state distribution of any mobility model based on random movement. We believe that the performance of our scheme will be more accurately evaluated under a mobility model with a steady-state distribution; accordingly, we will use RWM model with steady-state distribution obtained from RTM model.

3) *Attacker Models*: We assume that an adversary may compromise and fully control a subset of the sensor nodes, enabling him to mount various kinds of attacks. For instance, he can inject false data packets into the network and disrupt local control protocols such as localization, time synchronization, and route discovery process. Furthermore, he can launch denial of service attacks by jamming the signals from benign nodes.

To amplify his effectiveness, he can also launch a node replication attack, which is the subject of our investigation. We assume that the adversary can produce many replica nodes and that they will be accepted as a legitimate part of the network. We assume that the attacker attempts to employ as many replicas of one or more compromised sensor nodes in the network as will be effective for his attacks. We also assume that compromised and replica nodes follow the RWM model with the steady-state distribution when to move. We note that the attacker could move his replica nodes in different patterns in an attempt to frustrate our scheme. We discuss this possibility in Section III-B.

4) *Design Goals*: Under the above system and attacker models, we have three key design goals for replica detection. First, replica nodes should be detected with reasonable communication, computational, and storage overheads. Second, the detection schemes should be robust and highly resilient against an attacker's attempt to break the scheme. More specifically, the scheme should detect replicas unless the attacker compromises a substantial number of nodes. Finally, replica detection should be performed at the cost of minimal false positives and negatives. This is important to prevent turning the replica detection scheme into a tool for denial of service attacks.

III. MOBILE REPLICA DETECTION USING SEQUENTIAL PROBABILITY RATIO TEST

This section presents the details of our techniques to detect replica attacks in mobile sensor networks.

In static sensor networks, a sensor node can be considered to be replicated if it is placed at more than one location. However, if nodes are allowed to freely roam throughout the network, the above technique does not work because the mobile node's location will continuously change as it moves. Hence, it is imperative to use some other technique to detect replica nodes in mobile sensor networks. Fortunately, mobility provides us with a clue that can help resolve the mobile replica detection problem. Specifically, a mobile sensor node should never move faster than the system-configured maximum speed. Accordingly, if we observe that the mobile node's speed is over

the maximum speed, it is then highly likely that at least two nodes with the same identity are present in the network.

We propose a mobile replica detection scheme by leveraging this intuition. It is based on the Sequential Probability Ratio Test (SPRT) [25] which is a statistical decision process. SPRT has been proven to be the best mechanism in terms of the average number of observations that are required to reach a decision among all sequential and non-sequential test processes. SPRT can be thought of as one dimensional random walk with lower and upper limits [13], [26]. Before the random walk starts, null and alternate hypotheses are defined in such a way that the null one is associated with the lower limit and the alternate one is associated with the upper limit. A random walk starts from a point between two limits and moves toward the lower or upper limit in accordance with each observation. If the walk reaches or exceeds the lower or upper limit, it terminates and the null or alternate hypothesis is selected, respectively. We believe that SPRT is well suited for tackling the mobile replica detection problem in the sense that we can construct a random walk with two limits in such a way that each walk is determined by the observed speed of a mobile node; the lower and upper limits are properly configured to be associated with the shortfall and excess of the maximum speed of the mobile node, respectively.

We apply SPRT to the mobile replica detection problem as follows. Each time a mobile sensor node moves to a new location, each of its neighbors asks for a signed claim containing its location and time information and decides probabilistically whether to forward the received claim to the base station. The base station computes the speed from every two consecutive claims of a mobile node and performs the SPRT by taking speed as an observed sample. Each time maximum speed is exceeded by the mobile node, it will expedite the random walk to hit or cross the upper limit and thus lead to the base station accepting the alternate hypothesis that the mobile node has been replicated. On the other hand, each time the maximum speed of the mobile node is not reached, it will expedite the random walk to hit or cross the lower limit and thus lead to the base station accepting the null hypothesis that mobile node has not been replicated. Once the base station decides that a mobile node has been replicated, it initiates revocation on the replica nodes.

We first describe the detection scheme and then present analyses of its security and performance. Table I lists some of the notations used frequently in our discussion.

V_{max}	system-configured maximum speed of mobile sensor node.
N	total number of sensor nodes.
$ $	concatenation symbol.

TABLE I
NOTATIONS USED FREQUENTLY IN THIS PAPER

A. Protocol Description

Before deployment, every sensor node gets the secret keying materials for generating digital signatures. We will use an identity-based public key scheme such as [1], [7], [22]. It has demonstrated that public key operations can be efficiently implemented in static sensor devices [11], [16], [18]. Moreover, most replica detection schemes [8], [20], [28] employ an identity-based public key scheme for the static sensor networks. Mobile sensor devices are generally more powerful than static ones in terms of battery power due to the fact that the mobile sensor node consumes lots of energy to move. Additionally, the energy consumption for movement is known to be substantially larger than that for public key operations. For instance, the power consumption for the movement of a mobile sensor device was measured as 720 mW [9], whereas the energy consumption for public key signature is measured from 2.9 mW to 48 mW while that for public key verification was measured from 3.5 mW to 58.5 mW in accordance with the sensor hardware platforms [16]. Thus, we believe that the public key scheme can be practical for mobile sensor networks.

We also assume that every mobile sensor node is able to obtain its location information and verify the locations of its neighboring nodes. This can be implemented by employing GPS or secure localization methods such as those in [5], [12], [14], [15], [17]. This assumption may not lead to additional costs if the location information is used for other purposes. Finally, we assume that the clocks of all nodes are loosely synchronized with a maximum error of ϵ . This can be achieved by the use of secure time synchronization protocols such as [10], [23], [24]. Our proposed protocol proceeds in two phases.

1) *Claim Generation and Forwarding*: Each time a mobile sensor node u moves to a new location, it first discovers its location L_u and then discovers a set of neighboring nodes $N(u)$. Every neighboring node $v \in N(u)$ asks for an authenticated *location claim* from node u by sending its current time T to node u . Upon receiving T , node u checks whether T is valid or not. If $|T' - T| > \delta + \epsilon$ such that T' is the claim receipt time at u and δ is the estimated transmission delay of claim, then node u will ignore the request. Otherwise, u generates location claim $C_u = \{u || L_u || T || Sig_u\}$ and sends it to a neighboring node v , where Sig_u is the signature generated by node u 's private key. If u denies the claim request or if its claim fails to authenticate, then u will be removed from $N(v)$. Also, if u claims a location L_u such that the distance between L_v and L_u is larger than the assumed signal range of v , then it will be removed from $N(v)$. Once the above filtering process is passed, each neighbor v of node u forwards u 's claim to the base station with probability p .

2) *Detection and Revocation*: Upon receiving a location claim, the base station verifies the authenticity of the claim with the public key of node u and discards the claim if it is not authentic. We denote the authentic claims from node u by C_u^1, C_u^2, \dots . The base station extracts location information L_u^i and time information T_i from claim C_u^i . Let d_i denote the

Euclidean distance from location L_u^{i-1} at time T_{i-1} to L_u^i at T_i . Let o_i denote the measured speed at time T_i , where $i = 1, 2, \dots$. In other words, o_i is represented as:

$$o_i = \frac{d_i}{|T_i - T_{i-1}|} \quad (1)$$

Let S_i be denote a Bernoulli random variable that is defined as:

$$S_i = \begin{cases} 0 & \text{if } o_i \leq V_{max} \\ 1 & \text{if } o_i > V_{max} \end{cases}$$

The success probability λ of Bernoulli distribution is defined as:

$$\Pr(S_i = 1) = 1 - \Pr(S_i = 0) = \lambda \quad (2)$$

If λ is smaller than or equal to a preset threshold λ' , it is likely that node u has not been replicated. On the contrary, if $\lambda > \lambda'$, it is likely that node u has been replicated. The problem of deciding whether u has been replicated or not can be formulated as a hypothesis testing problem with null and alternate hypotheses of $\lambda \leq \lambda'$ and $\lambda > \lambda'$, respectively. In this problem, we need to devise an appropriate sampling strategy in order to prevent hypothesis testing from leading to a wrong decision. Specifically, we should specify the maximum possibilities of wrong decisions that we want to tolerate for a good sampling strategy. To do this, we reformulate the above hypothesis testing problem as one with null and alternate hypotheses of $\lambda \leq \lambda_0$ and $\lambda \geq \lambda_1$ such that $\lambda_0 < \lambda_1$, respectively. In this reformulated problem, the acceptance of the alternate hypothesis is regarded as a false positive error when $\lambda \leq \lambda_0$, and the acceptance of the null hypothesis is regarded as false negative error when $\lambda \geq \lambda_1$. To prevent the decision process from making these two types of errors, we define a user-configured false positive α' and false negative β' in such a way that the false positive and negative should not exceed α' and β' , respectively.

To understand the basis of this sampling plan, we present how SPRT is performed to make a decision about node u from the n observed samples, where a measured speed of u is treated as a sample. We first define the null hypothesis H_0 and the alternate one H_1 as follows: H_0 is the hypothesis that node u has not been replicated and H_1 is the hypothesis that u has been replicated. We then define L_n as the log-probability ratio on n samples, given as:

$$L_n = \ln \frac{\Pr(S_1, \dots, S_n | H_1)}{\Pr(S_1, \dots, S_n | H_0)} \quad (3)$$

Assume that S_i is independent and identically distributed. Then L_n can be rewritten as:

$$L_n = \ln \frac{\prod_{i=1}^n \Pr(S_i | H_1)}{\prod_{i=1}^n \Pr(S_i | H_0)} = \sum_{i=1}^n \ln \frac{\Pr(S_i | H_1)}{\Pr(S_i | H_0)} \quad (4)$$

Let ω_n denote the number of times that $S_i = 1$ in the n samples. Then we have

$$L_n = \omega_n \ln \frac{\lambda_1}{\lambda_0} + (n - \omega_n) \ln \frac{1 - \lambda_1}{1 - \lambda_0} \quad (5)$$

Where:

$$\lambda_0 = \Pr(S_i = 1 | H_0), \quad \lambda_1 = \Pr(S_i = 1 | H_1).$$

The rationale behind the configuration of λ_0 and λ_1 is as follows. λ_0 should be configured in accordance with the likelihood of the occurrence that benign node's speed exceeds V_{max} due to the time synchronization and localization errors. λ_1 should be configured to consider the likelihood of the occurrence that replica nodes' speeds exceed V_{max} .

On the basis of the log-probability ratio L_n , the SPRT for H_0 against H_1 is given as follows:

- $L_n \leq \ln \frac{\beta'}{1 - \alpha'}$: accept H_0 and terminate the test.
- $L_n \geq \ln \frac{1 - \beta'}{\alpha'}$: accept H_1 and terminate the test.
- $\ln \frac{\beta'}{1 - \alpha'} < L_n < \ln \frac{1 - \beta'}{\alpha'}$: continue the test process with another observation.

We can rewrite the SPRT as follows:

- $\omega_n \leq \tau_0(n)$: accept H_0 and terminate the test.
- $\omega_n \geq \tau_1(n)$: accept H_1 and terminate the test
- $\tau_0(n) < \omega_n < \tau_1(n)$: continue the test process with another observation.

Where:

$$\tau_0(n) = \frac{\ln \frac{\beta'}{1 - \alpha'} + n \ln \frac{1 - \lambda_0}{1 - \lambda_1}}{\ln \frac{\lambda_1}{\lambda_0} - \ln \frac{1 - \lambda_1}{1 - \lambda_0}}, \quad \tau_1(n) = \frac{\ln \frac{1 - \beta'}{\alpha'} + n \ln \frac{1 - \lambda_0}{1 - \lambda_1}}{\ln \frac{\lambda_1}{\lambda_0} - \ln \frac{1 - \lambda_1}{1 - \lambda_0}}$$

If a mobile node u is judged as benign node, the base station restarts the SPRT with newly arrived claims from u . If, however, u is determined to be replicated, base station terminates the SPRT on u and revokes all nodes with identity u from the network.

The pseudo-code for SPRT is presented as Algorithm 1.

B. Security Analysis

In this section, we will first describe the detection accuracy of our proposed scheme and then the limitations of replica node attacks when this scheme is employed.

1) *Detection Accuracy*: In SPRT, two types of errors are defined as follows:

- α : error probability that SPRT leads to accepting H_1 when H_0 is true.
- β : error probability that SPRT leads to accepting H_0 when H_1 is true.

Since H_0 is the hypothesis that a node u has not been replicated, α and β are the false positive and false negative probabilities of the SPRT, respectively. According to Wald's theory [25], we obtain the upper bounds of α and β as follows:

$$\alpha \leq \frac{\alpha'}{1 - \beta'}, \quad \beta \leq \frac{\beta'}{1 - \alpha'} \quad (6)$$

Furthermore, Wald proved that the sum of the false positive and negative probabilities of SPRT is limited by the sum of user-configured false positive and negative probabilities. Namely, the following equation holds:

$$\alpha + \beta \leq \alpha' + \beta' \quad (7)$$

Algorithm 1 SPRT for replica detection

INITIALIZATION: $n = 0$, $\omega_n = 0$
INPUT: location information L and time information T
OUTPUT: accept the hypothesis H_0 or H_1
 $\text{cur_loc} = L$
 $\text{cur_time} = T$
if $n > 0$ **then**
 compute $\tau_0(n)$ and $\tau_1(n)$
 compute speed o from cur_loc and prev_loc , cur_time
 and prev_time
 if $o > V_{max}$ **then**
 $\omega_n = \omega_n + 1$
 end if
 if $\omega_n \geq \tau_1(n)$ **then**
 accept the hypothesis H_1 and terminate the test
 end if
 if $\omega_n \leq \tau_0(n)$ **then**
 initialize n and ω_n to 0 and accept the hypothesis H_0
 return;
 end if
end if
 $n = n + 1$
 $\text{prev_loc} = \text{cur_loc}$
 $\text{prev_time} = \text{cur_time}$

Since β is the false negative probability, $(1 - \beta)$ is the replica detection probability. Accordingly, the lower bound on the replica detection probability will be:

$$(1 - \beta) \geq \frac{1 - \alpha' - \beta'}{1 - \alpha'} \quad (8)$$

From Equations 6 and 8, we can see that low user-configured false positive and negative probabilities will lead to a low false negative probability for the sequential test process. Hence, it will result in high detection rates. For instance, if the user configures α' and β' to both be 0.01, then replica detection is guaranteed with probability 0.99.

2) *Limitations of Replica Node Attacks:* We now discuss attacks that might be launched by the attacker and defenses provided by our scheme.

First, a malicious node v may attempt to forge a claim, either by sending a claim with incorrect data or by sending a claim with a bad signature. However, all of the v 's neighbors will check the validity of the v 's identity, reported location, reported time, and the signature over these values using node v 's public key. Alternatively, node v can simply ignore claim requests. In our scheme, if v 's benign neighbor does not receive a claim despite sending a claim request, it will remove v from its neighboring set and will not communicate with v .

We note that if one of v 's neighbors is malicious the malicious node can serve as v 's neighbor for forwarding packets. However, there is little benefit to the attacker of having a replica node in the same area as another compromised node. The compromised node can just as easily report fake data, participate in local control protocols, and eavesdrop on

messages sent through it. Further, if the attacker needs one compromised node to accompany each replica node in the network, there will be a very high cost for replica attacks. This assumption goes unstated, but is implied by the use of signed location claims in replica detection schemes for static networks as well [20], [8], [28].

Similarly, an attacker will not gain much benefit by forcing multiple replicas of a single node to form a group inside which each replica is close to each other whenever to move. This group mobility strategy substantially limits the region affected by his replicas and the attacker will not gain much from using the replicas in the limited region. For example, in a false data injection attack, it would be easy to ensure that only one of the replicas' data values at a time is accepted by the data aggregators. Similarly, in local control protocols, only one of the replicas' input values at a time would be taken by their neighbors. In this sense, multiple nodes with the same ID would not have more influence in a region than a single node.

Another approach for the attacker is to keep replicas close to each other so that the perceived velocity between their location claims is less than V_{max} . To do this, he coordinates a set of replicas to respond only to those claim requests that make it appear as a single node never moving faster than V_{max} . The attacker can have some replicas group closely together for this purpose; replicas that are further away must ignore claim requests to avoid detection. To illustrate this attacker's strategy, let us consider a simple attack scenario where a compromised node v and its replica v' are fixed to some locations in such a way that the distance between these two nodes is set to d . We assume that nodes v and v' initiate the neighbor discovery process at time T_0 and $T_0 + \frac{d}{V_{max}}$, respectively. Moreover, suppose that node v receives a claim request from neighbor x at time $T_0 + \xi$ and node v' receives request from neighbor y at time $T_0 + \frac{d}{V_{max}} + \xi$. Nodes v and v' send claims to x and y and ignore all incoming claim requests from other neighbors. Even though either x or y may move to a new location, the attacker can control nodes v and v' to accept claim requests from newly designated neighbors in such a way that the claim receipt time of v remains $\frac{d}{V_{max}}$ time ahead of that of v' . In this way, the attacker can successfully deceive the base station to believe that v moves back and forth with speed V_{max} . This attack scenario can be generalized to the case of a set of replicas and to allow for movement.

Since a set of replicas selectively respond to claim requests that help prevent them from being detected and discard the others, they can trick the base station into accepting H_0 , the hypothesis that they aren't replicas. Thus, a straightforward defense strategy against the above attack is to have the base station check whether each node responds to all incoming claim requests.

Specifically, we have the base station compute the claim receipt rate for each node, where the claim receipt rate is defined as the number of claims received by the base station per second. If the claim receipt rate for node v is below the pre-defined threshold, it is highly likely that v discards a substantial fraction of claim requests. In this case, the base station will

temporarily *quarantine* v from the network by disregarding all messages from v and broadcasting the quarantine information to all nodes. Upon receiving this quarantine message, all nodes will stop communicating with v . If the claim receipt rate for v increases to be above the pre-defined threshold during the quarantine period, the base station will release the quarantine that was imposed on v and broadcast the release information.

To more quickly decide whether or not to quarantine a node, we can replace the static threshold approach with SPRT as follows. Let r_v be denote a claim receipt rate for node v measured by the base station. Moreover, let us define μ as the mean of the claim receipt rates for all nodes and let I_v be a Bernoulli random variable defined as:

$$I_v = \begin{cases} 0 & \text{if } r_v \geq \rho\mu \\ 1 & \text{if } r_v < \rho\mu \end{cases}$$

Where: $0 < \rho < 1$

Now we define H_0 and H_1 as follows: H_0 is the hypothesis that node v responds to all incoming claim requests, and H_1 is the hypothesis that node v responds to only a limited fraction of incoming claim requests. The base station performs SPRT for node v with the above definitions. If it accepts H_1 , it quarantines on v . If the decision result of v is changed to H_0 during the quarantine period, it releases the quarantine on v . This process is repeated in accordance with the decision result. To prevent oscillation between quarantine and non-quarantine states, we can require the quarantined node to remain in H_0 for some minimum period of time ΔT . ΔT can be set to ensure that the replica nodes would be quarantined for longer periods than they would be able to participate freely in the network.

C. Performance Analysis

We now present a performance analysis of our scheme in terms of communication, computation, and storage overheads.

1) *Communication Overhead*: We first describe how many observations on an average are required for the base station to make a decision on whether a node has been replicated or not. Then we will present the communication overhead of our scheme.

Let n denote the number of samples to terminate SPRT. Since n varies with the types of samples, it is treated as a random variable with an expected value $E[n]$. According to the Wald's theory [25], we can obtain $E[n]$ as follows.

$$E[n] = \frac{E[L_n]}{E \left[\ln \frac{\Pr(S_i|H_1)}{\Pr(S_i|H_0)} \right]} \quad (9)$$

From this equation, we compute the expected numbers of n conditioned on the hypotheses H_0 and H_1 as follows:

$$E[n|H_0] = \frac{(1 - \alpha') \ln \frac{\beta'}{1 - \alpha'} + \alpha' \ln \frac{1 - \beta'}{\alpha'}}{\lambda_0 \ln \frac{\lambda_1}{\lambda_0} + (1 - \lambda_0) \ln \frac{1 - \lambda_1}{1 - \lambda_0}}$$

$$E[n|H_1] = \frac{\beta' \ln \frac{\beta'}{1 - \alpha'} + (1 - \beta') \ln \frac{1 - \beta'}{\alpha'}}{\lambda_1 \ln \frac{\lambda_1}{\lambda_0} + (1 - \lambda_1) \ln \frac{1 - \lambda_1}{1 - \lambda_0}} \quad (10)$$

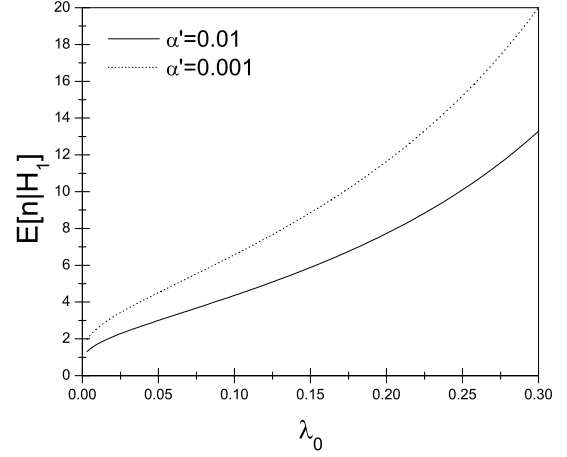


Fig. 1. λ_0 vs. $E[n|H_1]$ when $\lambda_1 = 0.7$ and $\beta' = 0.01$.

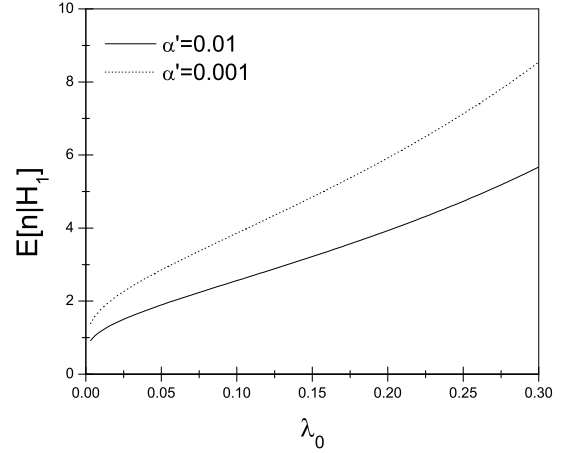


Fig. 2. λ_0 vs. $E[n|H_1]$ when $\lambda_1 = 0.9$ and $\beta' = 0.01$.

We study how $E[n|H_1]$ is affected by the values of λ_0 and α' . As shown in Figures 1 and 2, $E[n|H_1]$ tends to increase as the rise of λ_0 when λ_1 is fixed to 0.7 and 0.9, respectively. This implies that the small value of λ_0 contributes to detect replicas with the small number of claims. When λ_0 is fixed, $E[n|H_1]$ in the case of $\lambda_1 = 0.7$ is larger than the corresponding value in the case of $\lambda_1 = 0.9$. We infer from this that the large value of λ_1 reduces the number of claims required for replica detection. Finally, we see that the number of claims in the case of $\alpha' = 0.001$ is more than that for $\alpha' = 0.01$ in both Figures 1 and 2. This indicates that we will need to increase of the number of claims for replica detection if we want to reduce the user-configured false positive.

Now we compute the communication overhead of our scheme. We define communication overhead as the average number of claims that are sent or forwarded by nodes in the

network. Each time a mobile node u receives b claim requests on average at a location, it sends an average of $b \times p$ claims to the base station, where p is the probability that the claim is forwarded to the base station. Let us consider the worst scenario where every mobile node receives b claim requests at a location and sends $b \times p$ claims to the base station at the same time. Since the average hop distance between two randomly chosen nodes is given by $O(\sqrt{N})$ [20] where N is the total number of sensor nodes, the communication overhead of the worst case will be $O(b \times p \times N \times \sqrt{N})$. Each node's b claim requests contain the same location information L . Indeed, the base station needs only one claim per location L . In this sense, $b \times p$ can be reduced to one by setting p to $\frac{1}{b}$. Thus, the communication overhead in the worst scenario can be rewritten as $O(N\sqrt{N})$ and is equivalent to one of the best scheme of [20].

2) *Computation and Storage Overhead*: We define computation and claim storage overhead as the average number of public key signing and verification operations per node and the average number of claims that needs to be stored by a node, respectively.

Each time a mobile node receives b claim requests on average at a location, it needs to perform b signature generation operations. Similarly, each time a mobile node sends b claim requests on average at a location, it needs to verify up to b signatures. In the worst case, every mobile node sends $b \times p$ claims to the base station at the same time, the base station thus needs to verify up to $b \times p \times N$ signatures. If p is set to $\frac{1}{b}$, the base station will verify up to N signatures on average in the worst case.

The base station stores location claims in order to perform SPRT whereas every sensor node does not need to keep its or other nodes' claims. Thus, we only need to compute the number of claims that are stored by the base station. In SPRT, a sample o_i is obtained from two consecutive location claims of node u , namely C_u^{i-1} and C_u^i . Once a sample o_i is obtained, the previous location claim C_u^{i-1} is discarded and current location claim C_u^i is maintained by the base station. This process is repeated until SPRT is terminated. Hence, the base station needs to store only one claim per node, so at most N claims are required to be stored in the base station.

IV. SIMULATION STUDY

In this section, we will first describe the simulation environment and then discuss the simulation results.

A. Simulation Environment

We simulated the proposed scheme by using the ns-2 network simulator. In our simulation, 500 mobile sensor nodes are placed within a square area of 500 m \times 500 m.

We use Random Waypoint Mobility (RWM) model to determine movements of mobile sensor nodes. In the RWM model, each node moves to a randomly chosen location with a randomly selected speed between a predefined minimum and maximum speed. After reaching that location, it stays there for a predefined pause time. It then randomly chooses another

location after that pause time and moves to that location. This random movement process is repeated during a simulation time. As mentioned in Section II, we use the RWM model with the steady-state distribution provided by Random Trip Mobility (RTM) model. Specifically, it is generated by the code of the RTM model that was implemented in [19].

All simulations were performed for 1000 simulation seconds. We fixed a pause time of 20 simulation seconds and a minimum moving speed of 1 m/s of each node. Each node uses IEEE 802.11 as the media access control (MAC) protocol in which the transmission range is 50 m. We set the both the user-configured false positive threshold α and the false negative threshold β to 0.01, and we set λ_0 and λ_1 to 0.01 and 0.7, respectively. The rationale behind these configurations has been discussed in Section III.C.

In our simulation, we consider one benign node, one compromised node and its replica node as claim generators. Furthermore, these three nodes' initial placements are randomly chosen and their movements are randomly determined by the RWM model with a steady-state distribution. We assume that all claims that have been forwarded to the base station reach it without any loss. We also assume that both time synchronization and localization protocols perfectly work without incurring any error. We repeated the simulation 1000 times in such a way that every mobile node is initially placed in a different random location each time.

B. Simulation Results

We use the following metrics to evaluate the performance of our scheme:

- *Number of Claims* is the number of claims required for the base station to decide whether a node has been replicated or not.
- *False Positive* is the error probability that a benign node is misidentified as replica node.
- *False Negative* is the error probability that a replica node is misidentified as benign node.

For each execution, we obtain each metric as the average of the results of SPRTs that are repeated. Note that SPRT will be terminated if it decides that the claim generator has been replicated. The average of the results of 1000 executions is presented.

In the experiment, the average number of requests b was measured from 17 to 20 in accordance with V_{max} . We associate the configuration of claim forwarding probability p with b . Specifically, p is configured to 0.05 in order to set $b \times p$ to 1 when b is assumed to be 20 at all V_{max} . The rationale behind this configuration is to make sure that one claim per location is forwarded to the base station on average.

We investigate the false positive and false negative rates, and the number of claims while varying the maximum movement speed V_{max} from 10 m/s to 100 m/s. The results are summarized as follows.

First, there were no false positives or false negatives at all mobility rates. This implies that the replica was always detected with probability 1 and the benign node was never

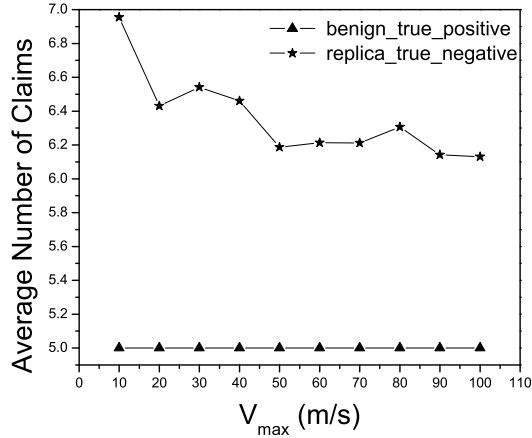


Fig. 3. V_{max} vs. Average Number of Claims.

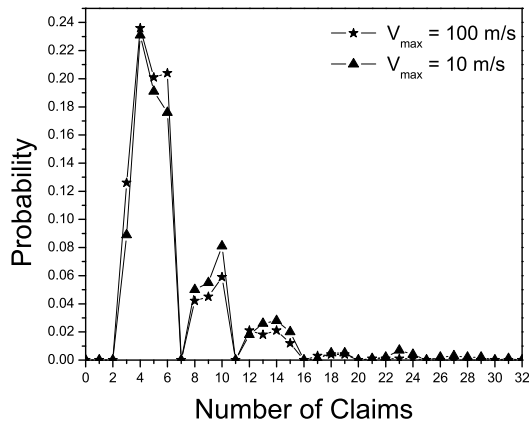


Fig. 4. Probability distribution of the number of claims.

misidentified as a replica at any mobility rate. From this observation, we see that SPRT works well against our random attacker model.

Second, the results of the average number of claims are shown in Figure 3. We present the results of two cases. One is that the claim generator is a benign node and the SPRT decides that this node is a benign node. We denote this case by *benign_true_positive* in Figure 3. The other one is that the claim generators are a compromised node and its replica node, and SPRT decides that these nodes are a compromised node and its replica. We denote this case by *replica_true_negative* in Figure 3.

In the case of *benign_true_positive*, the average number of claims is 5.0 whereas the theoretical expected value of claims is 3.95. In the case of *replica_true_negative*, the average number of claims is at most 6.96. Since two nodes contributed

to the average of this case, the average per node becomes at most 3.48 whereas the theoretical expected value of claims is 1.72. In both cases, the simulation results are higher than the theoretical ones. This implies that the claims containing the same location but different time contribute to the raise of the average number of claims. Even though there is a gap between the simulation and theoretical results, the base station still reaches correct decisions with very few claims.

We also see that the average number of claims is the same regardless of the mobility rate in the case of *benign_true_positive*. Since we assume error-free time synchronization and localization protocols, the benign node's speed never exceeded V_{max} at any mobility rate and thus the same number of observations always moves the test toward H_0 . In the case of *replica_true_negative*, the average number of claims tends to slightly decrease as the mobility rate increases. This is mainly because the increase in mobility leads to a reduction in the chance that the node generates the claims containing the same location but different time, and thus expedites moving the test toward H_1 .

Finally, Figure 4 shows the probability distribution of the number of claims in the case of *replica_true_negative*. For this distribution, we consider two cases of low ($V_{max} = 10$ m/s) and high ($V_{max} = 100$ m/s) mobility rates. A total of 68.7% and 76.7% of the cases fall in the range from 3 to 6 claims in the case of low and high mobility rates, respectively. This implies that in most cases, the number of claims is less than the average and thus SPRT detects replicas in fewer than six claims in most cases.

V. RELATED WORK

The first work on detecting replica attacks is due to Parno et al. [20], who proposed randomized and line-selected multicast schemes to detect replicas in *static* wireless sensor networks. In the randomized multicast scheme, every node is required to multicast a signed *location claim* to randomly chosen witness nodes. A witness node that receives two conflicting location claims for a node concludes that the node has been replicated and initiates a process to revoke the node. The line-selected multicast scheme, on the other hand, reduces communication overhead of the randomized multicast scheme by having every claim-relaying node participate in the replica detection and revocation process.

Conti et al. [8] proposed a new method for selecting witness nodes for replica detection. This method enhances the line-selected multicast scheme of [20] in terms of replica detection probability, storage and computation overheads. Zhu et al. [28] proposed localized multicast schemes based on grid cell topology. In [28], replicas are detected by letting location claim be multicast to single cell or multiple cells. The main strength of [28] is that it has higher detection rates and lower storage overheads than the best scheme of [20].

Choi et al. [6] proposed a clone detection scheme in which the sensor network was virtually divided into a set of non-overlapping subregions. An exclusive subset is formed in each subregion. If the intersection of subsets is not empty, it implies

that replicas are included in those subsets. However, this scheme allows an adversary to bypass the detection by placing replicas in such a way that any two adjacent subsets do not include them.

However, none of these solutions is suitable for replica node detection in mobile sensor networks. If the location claim approach is used in mobile sensor networks, sensor nodes' location claims will be continuously changed in accordance with their movements, and location claims from the same node will always conflict each other. If the scheme in [6] is used in mobile sensor networks, nodes' movement will reduce the probability that replicas are included in the intersection of exclusive subsets.

VI. CONCLUSIONS

We have proposed a replica detection scheme for mobile sensor networks based on the Sequential Probability Ratio Test (SPRT). We perform SPRT to detect mobile replicas by using the basic idea that a mobile node should never move at speeds in excess of the system-configured maximum speed. Our scheme quickly detects mobile replicas with very small number of location claims. Furthermore, we have presented two types of attacks that might be launched by the attacker and discussed the defense strategies against those attacks.

For future work, we would like to thoroughly explore how localization and time synchronization errors affect the detection accuracy of our scheme. We will examine analytically and in simulation how time synchronization and localization errors influence the false positive and negatives of the scheme. Furthermore, we would like to evaluate our scheme against various types of attacker models. In particular, we are interested in exploring how a variety of attacker models impact on the security of the scheme. We believe that a game theoretic model is suited for investigating the interactions between the detector and the adversary. In this model, we will first study a variety of strategies that may be taken by detector and adversary. Then, we will try to find the Nash Equilibrium for those strategies.

REFERENCES

- [1] D. Boneh and M.K. Franklin. Identity-Based Encryption from the Weil Pairing. In *Advances in Cryptology CRYPTO*, 2001.
- [2] J-Y. L. Boudec and M. Vojnović. Perfect Simulation and Stationary of a Class of Mobility Models. In *IEEE INFOCOM*, 2005.
- [3] J.B. Broch, D.A. Maltz, D.B. Johnson, Y-C. Hu, and J.G. Jetcheva. A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols. In *ACM MobiCom 1998*, October 1998.
- [4] T. Camp, J. Boleng, and V. Davies. A survey of mobility models for ad hoc network research. *Wireless Communication and Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends, and Applications*, 2(5):483-502, 2002.
- [5] S. Čapkun and J.P. Hubaux. Secure Positioning in Wireless Networks. *IEEE Journal on Selected Areas in Communications*, 24(2):221–232, February 2006.
- [6] H. Choi, S. Zhu, and T.F La Porta. SET: Detecting node clones in Sensor Networks. In *IEEE/CreateNet Conference on Security and Privacy for Emerging Areas in Communication Networks (SecureComm)*, 2007.
- [7] C. Cocks. An Identity Based Encryption Scheme Based on Quadratic Residues. In *the 8th IMA International Conference on Cryptography and Coding*, 2001.
- [8] 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 *ACM Mobihoc*, 2007.
- [9] K. Dantu, M. Rahimi, H. Shah, S. Babel, A. Dhariwal, and G. S. Sukhatme. Robomote: enabling mobility in sensor networks In *IEEE IPSN*, 2005.
- [10] S. Ganeriwal, S. Čapkun, C.C. Han, and M.B. Srivastava. Secure time synchronization service for sensor networks. In *ACM WiSe*, 2005.
- [11] V. Gupta, M. Millard, S. Fung, Y. Zhu, N. Gura, S. Eberle, and H. Chang. Sizzle: A Standards-based End-to-End Security Architecture for the Embedded Internet. In *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, March 2005.
- [12] L. Hu and D. Evans. Localization for Mobile Sensor Networks. In *ACM Mobicom*, 2004.
- [13] J. Jung, V. Paxon, A.W. Berger, and H. Balakrishnan. Fast port detection using sequential hypothesis testing. In *IEEE Symposium on Security and Privacy*, 2004.
- [14] L. Lazos, S. Čapkun, and R. Poovendran. ROPE: Robust Position Estimation in Wireless Sensor Networks. In *IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, April 2005.
- [15] Z. Li, W. Trappe, Y. Zhang, and B. Nath. Robust Statistical Methods for Securing Wireless Localization in Sensor Networks. In *IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, April 2005.
- [16] A. Liu and P. Ning. TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks. In *Technical Report TR-2007-36, North Carolina State University, Department of Computer Science*, November 2007.
- [17] D. Liu, P. Ning, and W. Du. Attack-Resistant Location Estimation in Sensor Networks. In *IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, April 2005.
- [18] D. Malan, M. Welsh, and M. Smith. A public-key infrastructure for key distribution in TinyOS based on elliptic curve cryptography. In *IEEE Conference on Sensor and Ad hoc Communications and Networks (SECON)*, October 2004.
- [19] S. PalChaudhuri, J-Y. L. Boudec, and M. Vojnović. Perfect Simulations for Random Trip Mobility Models. In *38th Annual Simulation Symposium*, April 2005.
- [20] B. Parno, A. Perrig, and V.D. Gligor. Distributed detection of node replication attacks in sensor networks. In *IEEE Symposium on Security and Privacy*, May 2005.
- [21] J. Reich and E. Sklar. Robot Sensor Networks for Search and Rescue. In *IEEE International Workshop on Safety, Security and Rescue Robotics*, 2006.
- [22] A. Shamir. Identity-Based Cryptosystems and Signature Schemes. In *Advances in Cryptology CRYPTO*, 1984.
- [23] H. Song, S. Zhu, and G. Cao. Attack-resilient time synchronization for wireless sensor networks. *Ad Hoc Networks*, 5(1):112–125, January 2007.
- [24] K. Sun, P. Ning, C. Wang, A. Liu, and Y. Zhou. TinySeRSync: Secure and Resilient Time Synchronization in Wireless Sensor Networks. In *ACM CCS*, 2006.
- [25] A. Wald. *Sequential Analysis*. Dover Publications, 2004.
- [26] M. Xie, H. Yin, and H. Wang. An Effective Defence Against Email Spam Laundering. In *ACM CCS*, 2006.
- [27] J. Yoon, M. Liu, and B. Noble. Random waypoint considered harmful. In *IEEE INFOCOM*, 2003.
- [28] B. Zhu, V.G.K. Addada, S. Setia, S. Jajodia, and S. Roy. Efficient Distributed Detection of Node Replication Attacks in Sensor Networks. In *ACSAC*, December 2007.