

Wormholes no more?

Localized Wormhole Detection and Prevention in Wireless Networks

Tassos Dimitriou and Athanassios Giannetsos

Athens Information Technology,
19002, Athens, Greece
{tdim, agia}@ait.edu.gr

Abstract. A number of protocols have been proposed to date to defend against *wormhole attacks* in wireless networks by adopting synchronized clocks, positioning devices, or directional antennas. In this work, we introduce a novel approach for detecting wormhole attacks. The proposed algorithm is completely *localized* and works by looking for simple evidence that no attack is taking place, using only connectivity information as implied by the underlying communication graph, and total absence of coordination. Unlike many existing techniques, it does not use any specialized hardware, making it extremely useful for real-world scenarios. Most importantly, however, the algorithm can *always* prevent wormholes, irrespective of the density of the network, while its efficiency is not affected even by frequent connectivity changes. We also provide an analytical evaluation of the algorithm's correctness along with an implementation on real sensor devices that demonstrates its efficiency in terms of memory requirements and processing overhead.

Key words: Wormhole attack, Sensor Networks, Ad hoc networks, Computer network security, Tunnelling, Connectivity Information, Path Existence

1 Introduction

The use of wireless ad hoc and sensor networks has given birth to a broad class of exciting new applications in several areas of our lives. However, these networks are exposed to security threats which, if not properly addressed, can exclude them from being deployed in the envisaged scenarios.

The *wormhole attack* [7] is a severe threat against wireless networks in which an adversary tunnels messages received in one part of the network and replays them in a different part, as shown in Figure 1(a). Once the wormhole link is operational, the adversary eavesdrops messages at one end and forwards them (possibly selectively) to the other end, where the packets are retransmitted. Thus nodes who would normally be multiple hops away from a sink are convinced that they are only one or two hops away via the wormhole.

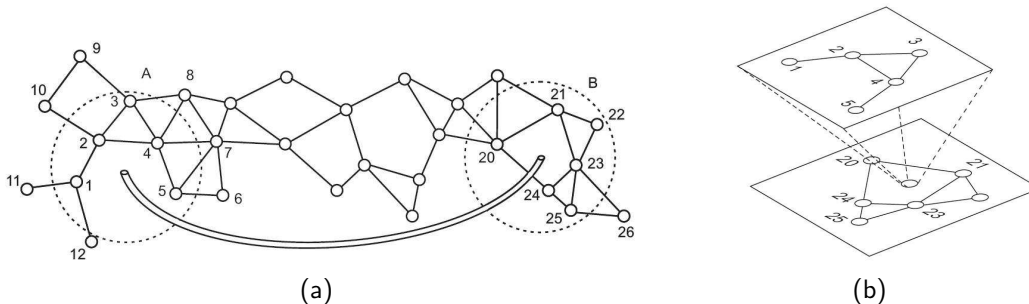


Fig. 1. (a) Demonstration of a wormhole attack. Nodes in area *A* consider nodes in area *B* their neighbors and vice versa. (b) Projection of the two ends of the wormhole.

The net effect of the wormhole attack is that nodes within region *A* think they are neighbors with nodes within region *B* and vice versa. This is equivalent with taking all nodes in one area and place them at just one point within another area, as shown in Figure 1(b). Thus an attacker can use it to completely disrupt routing and attract a significant amount of traffic enabling other kind of attacks such as the Sinkhole attack [7, 10].

Our contribution: In this paper, we explore the development of a *localized* algorithm that can detect wormhole attacks on wireless networks directly based on *connectivity* information implied by the underlying communication graph. Our work deviates from the customary strategies of using specialized hardware in sensors, directional antennas, tight clock synchronization, or distance measurements between the nodes, thus making this approach universally applicable.

The detection algorithm searches for simple structures that indicate that no attack is taking place and its efficiency is not affected even by frequent connectivity changes, another deviation from past works that assume *static* topologies. We provide an analytical evaluation of the algorithm’s performance and correctness showing that attack prevention is 100% (even for low density networks) while keeping the running time and the percentage of false positives very small. Finally, we present an implementation of the algorithm on real sensor devices, justifying its efficiency in terms of memory requirements and processing overhead. In general, we expect that our approach will have a practical use in sensor network deployments overcoming some of the limitations of existing techniques.

Paper Organization: The remainder of this paper is organized as follows. First, we discuss related work in Section 2 and state our assumptions in Section 3. Section 4 is the heart of this work; it gives insight on the algorithm, a mathematical proof about its correctness in preventing wormhole attacks and a probabilistic analysis on its behavior on legitimate node addition. Performance evaluation, in terms of simulations and experiments on real sensor devices, is presented in Section 5. Section 6 sets forth some attributes of our algorithm for discussion, while Section 7 concludes the paper.

2 Related Work

In this section we briefly describe the most widely known proposed measures for mitigating the effects of a wormhole attack in wireless networks. Enhancing routing with strong node authentication and lightweight cryptography [7] was proposed as a countermeasure, however, the wormhole attack still cannot be defeated that easily as an attacker can simply forward existing packets.

An alternative set of mechanisms that operate independently from the underlying routing protocol is based on *distance/time bounding* techniques. In [5], the authors add a secure constraint (leash) to each packet such as timing or location information that used to tell whether a packet has traveled a distance larger than physically possible. This technique works fine in the presence of specialized hardware for localization and synchronization, however this assumption raises questions about its applicability to ordinary sensor networks.

Another set of solutions use authenticated *distance bounding* of the round trip time (RTT) of a message [3, 8, 15], received signal strength, or time difference of arrival [16] in order to ensure that nodes are close to each other. The effectiveness of such schemes relies on the immediate reception of responses to challenges sent, however, this may not be possible as MAC protocols introduce random delays between the time a packet is sent and the actual time it is transmitted via the radio interface. In [14] the authors proposed a solution that assumes the existence of a small fraction of nodes that are aware of their own location. This approach is showed to successfully detect a wormhole with probability close to one. However, its functionality relies on the correct operation of distinguished guard nodes that can become single points of failure.

Another line of defense is the use of *graph theoretic and visualization* approaches. In [17], a centralized detection technique is proposed which uses distance estimations between neighboring nodes in order to determine a “network layout” and identify inconsistencies in it, using multi-dimensional scaling (MDS) techniques. However, its centralized nature limits its applicability in sensor networks. Similarly the works in [14, 6] make use of guard nodes that attest the source of each transmission. However, these techniques either make location claims or use special purpose hardware making these approaches impractical.

Finally, another interesting approach to date is the work in [13] which looks for topological violations in the underlying connectivity graph in order to detect a wormhole attack. For example, in Figure 1(b), nodes 1, 3 and 5 cannot possibly be neighbors of (say) 20 and 23 since it can be shown that three 2-hop neighbors cannot be the common neighbors of two other nodes u and v . This is an instance of *forbidden substructure* and this is exactly the approach followed in [13] for detecting a wormhole.

While this is a nice, localized approach that uses connectivity information to detect a wormhole attack it suffers from a number of shortcomings. First, it does not always guarantee detection since existence of three 2-hop neighbors lying in the common intersection area of two others depends on the density of the network. The technique will probably fail when the average neighborhood size is low. To alleviate this problem one may look for similar forbidden substructures

of size f_k among k -hop neighbors of u and v . Unfortunately, there are two issues with this approach. The first one is that the forbidden substructure is really an *independent set* of the set of common $2k$ -hop neighbors $C_{2k}(u, v)$, a known NP-complete problem. Of course one may try to find a maximal independent set in $C_{2k}(u, v)$ and try to compare it with f_k . If the size of the independent set is equal or greater than f_k then an alarm can be raised. However, the value of f_k for $k > 1$ cannot be estimated that easily. For example, for $k = 2$ (2-hop case) this number is as big as 19 [13]. Unless the node density is extremely high, it is unlikely that one will be able to find that many common independent 2-hop neighbors in order to detect the attack.

3 System Model and Assumptions

3.1 Sensor Nodes and Communication

In our model, a wireless sensor network consists of a set $S = \{s_1, s_2, \dots, s_n\}$ of n sensor nodes. Sensor nodes are considered neighbors when the distance between them is shorter than some range r . For any sensor node s , the set of neighboring nodes is denoted by $N(s)$. Our first assumption concerns neighborhood information and is denoted by SMA-1 (for System Model Assumption):

SMA-1 All sensors run some neighbor discovery routine, as part of their routing protocol (e.g. using regular route updates), and they can record their neighbor IDs. Thus every node knows its k -hop neighborhood, where k is a small number, typically 1 or 2.

The above assumption is light and realistic, considering the case of sensor networks with frequent neighborhood changes¹. Furthermore, a lot of research in such networks has been conducted based on the 2-hop neighborhood knowledge, covering areas from energy consumption efficiency [11, 12] to intrusion detection [4, 9]. In any case, as we will see shortly and discuss further in Section 6, even if SMA-1 does not hold, *no wormholes can be allowed in the network*.

3.2 Attacker Model

Consider an adversary that tries to mound a wormhole attack. The wormhole is a dedicated connection between two physical locations, called *wormhole endpoints*. By re-transmitting packets, an adversary can have nodes hear each other and establish a neighbor relationship, even if they do not reside in each other's radio range. Below is what we have assumed to better model the wormhole attack:

AMA-1 In this work, we will consider attacks in which the wormhole link is long enough so that regions A and B are well separated from each other [2, 19].

¹ Notice that this assumption does not say anything about the *validity* of these neighbor IDs. Only that these IDs are forwarded properly by the routing protocol.

Thus it makes no sense to have overlapping endpoints or endpoints close to each other. In particular, we will assume that the real shortest path distance between the two wormhole regions is bigger than $2k$ hops, where k (usually 1 or 2) denotes the k -hop neighborhood structure known to nodes.

We place no restrictions on what an adversary can do with packets that carry neighborhood information. The adversary can drop these packets, however, even in this case, no wormholes can be allowed in the network. In the sections that follow we will see that both assumptions SMA-1 and AMA-1 are not necessary to the correctness of this work. However, the following assumption is:

AMA-2 There is some initial interval t_Δ where no attack has taken place and nodes have safely established their neighborhood information.

This assumption simply says that there must be some initial interval where the network is safe in order for the algorithm to guarantee prevention of wormhole attacks from that time on. This is a standard assumption made in many of the works in this area [2, 13, 14, 17–19] and more general in works where some sort of security infrastructure has to be bootstrapped.

4 Localized Wormhole Detection and Prevention

Our approach is strictly *localized* and looks only at *connectivity* information as implied by the underlying communication graph. However, instead of looking for forbidden substructures as in [13], we look for evidence that no attack is taking place. This has some interesting consequences.

- Our approach is applicable even when the communication model is unknown or the network is deployed in an ad-hoc manner.
- Second, in the case of an attack, the algorithm always prevents the attack.
- Third, our algorithm “fails safe”; in the unlikely case where this desired property is not met, the algorithm treats suspicious nodes as participating in an attack. Thus, no wormhole can ever be established. This is in contrast with [13] where wormhole detection cannot be guaranteed in all cases.
- Finally, as we will see in Section 5, the algorithm is very easy to be implemented, even in resource constrained devices such as sensor nodes.

To understand the workings of the algorithm, let’s assume that up to time $t \geq t_\Delta$ the network is “safe” (i.e. no wormholes have occurred – Assumption AMA-2) and nodes know each other’s k -neighborhoods (Assumption SMA-1).

At some time $t' > t$ a number of neighboring nodes in a set $U = \{u_1, u_2, \dots\}$ overhear some packets transmitted that include the IDs of new nodes in a set $V = \{v_1, v_2, \dots\}$. These packets may or may not be the result of a wormhole attack. For example, they may be *newly added* nodes or simply nodes that awoke from a sleeping phase and participate in the workings of the underlying routing protocol. Or in the case of wormhole attack (Figure 1), they may be retransmissions of packets from one area to another. We call the nodes in V *suspected* nodes.

Each node $u \in U$ must determine for each node $v \in V$ whether it should include v in its neighborhood structure. This is done using the following test.

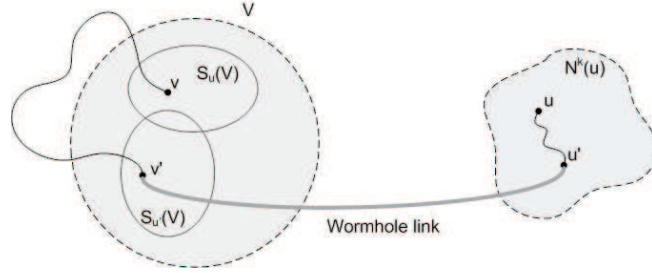


Fig. 2. Contradiction argument

4.1 Local Path Existence Test for Wormhole Prevention

The *localized* test run by each node $u \in U$ for each node $v \in V$ is to determine whether a *small* path (of length no more than $2k$) exists that connects u to v but in a way that *excludes all suspect nodes*. This is possible since u also knows the k -hop neighbors of v . This is done by considering whether u and v have a common neighbor lying in their intersection or two neighbors x and y , respectively, that are either directly connected to each other or have a common 1-hop neighbor z . In the more general case, x and y can either be directly connected or have a common $(k - 1)$ -hop neighbor. This alternative path, if it exists, will be an attestation that no wormhole link exists and u can safely add v in its neighborhood list. Otherwise v is deleted from the neighborhood of u . This is captured by the following theorem:

Theorem 1. *In case of a wormhole attack, the previous test prevents a wormhole link from being established. If no attack is taking place, the algorithm allows new nodes to be part of the network with high probability.*

Proof. For a formal proof we will consider two cases: i) the case of an actual wormhole attack, and ii) the case of legitimate addition of new nodes.

Case 1: Let $u \in U$ be a node wishing to test whether some suspect node $v \in V$ should be included in its neighborhood list. Let $D > 2k$ be the *real* shortest path distance between u and v in the network (assumption AMA-1 on separation of wormhole endpoints). Let also $N^k(v)$ denote the set of *valid* k -hop neighbors of v also known to u (Assumption AMA-2), and let $S_u(V)$ denote the nodes suspected by u from the set V . We place no restriction on the suspect set, so neither all nodes in U may suspect the same set of nodes, nor they have to agree on a common suspect list which would raise synchronization issues.

Consider Figure 2 showing the set of k -hop neighbors of u on the right and the suspect node v on the left. Since u *itself* will check for the existence of a path between u and v of length no more than $2k$, the only way such a path can exist is if the path at some point utilizes the wormhole link and two nodes u' and v' that are at most k hops away from u and v , respectively. If the path from v uses only nodes in $S_u(V)$, u will easily reject such a path. The problem arises when the path uses nodes not in $S_u(V)$. There are two cases here to consider.

The first case is when the path consists of nodes outside V . But in this case u will reject all these paths since their length is at least D and they can never reach u in at most $2k$ hops. The second case is when the path uses a mix of nodes that either belong to V or not. In such a case all these paths must end with some node $v' \in V - S_u(V)$. The path from v' will utilize the wormhole link and will try to close the loop using some k -hop neighbor u' of u . There are two cases again to consider. Either $v' \in S_{u'}(V)$ or not.

In the first case, u' overheard a packet containing the ID of v' and placed v' in a quarantine (suspect list). But then it cannot have moved v' in its neighborhood list unless it had performed a “small path” test with v' . Thus u will never be fooled in accepting v' as a legitimate neighbor of u' since it always has up-to-date information regarding the neighborhood structure of u' . In the second case, u' never heard anything about v' so it definitely does not have v' in its neighbors list. Thus again u will reject the path. In summary, no wormhole can be established between the sets U and V .

Case 2: Consider now the benign case where no attack takes place (V may be a set of newly added nodes). Node u will attempt to establish a small path with v , excluding the suspect nodes. This can happen in a straightforward way and we leave the detailed description for Section 4.3.

We need to point out, however, that the algorithm may fail to find short paths² and as a result treat these nodes as part of a wormhole attack. This is an instance of the “safe failure” principle we highlighted in the beginning of this section. *We opted for preventing a wormhole link from being established when an attack is taking place at the expense of not allowing some legitimate nodes to be part of the network when no attack occurs.* However, as we will demonstrate probabilistically in the next section and verify experimentally in Section 5, the probability of this happening is very small even for low density networks.

4.2 Existence of Short Paths - Probabilistic Analysis

We will now argue about the existence of small paths between two nodes u and v . We model the topology of the wireless sensor network by a random geometric graph which can be constructed as follows: we throw n nodes uniformly at random onto the surface of a unit square and we connect all nodes within distance r of each other. An edge (u_i, u_j) in the geometric graph corresponds to a communication link between sensors i and j . The analysis will be performed in terms of the following quantities (that are either given or easily derived):

- the number of sensors in the field, n ;
- the communication range, r ;
- the probability p that two random nodes u and v are neighbors;
- the average density d of the graph, i.e. the expected number of neighbors of a given node.

² Either because of lack of common neighbors with v or because some neighborhood updates were lost.

In what follows, we will express all results in terms of d . If u is a node with range r in the unit square, the probability p that another node v is a neighbor of u is given by the quantity $p = \pi r^2$. Thus the expected number of neighbors d of u is equal to $d = (n - 1)p \approx np = n\pi r^2$, for large enough n .

Our goal is to upper bound the probability that no path exists between u and v when 2-hop neighborhoods are known. It can be shown (the proof is omitted due to space restrictions) that

$$\Pr[\text{No small path between } u \text{ and } v] < e^{-0.58d},$$

where d is the average network density. Thus the probability that u and v will be connected is lower bounded by $1 - e^{-0.58d}$. This probability is very high even for low density networks (more than 94% when $d = 5$ and more than 99% when $d = 10$). And as we will see in Section 5, this probability increases even further when we consider paths between 1-hop neighbors of u and 1-hop neighbors of v .

4.3 Algorithm Description

We will now present in more details the Wormhole Detection Algorithm (WDA) described in the previous sections. Recall that WDA is to look for a simple path between two nodes u and v that indicates absence of a wormhole link between them. The algorithm is strictly *localized* and therefore only nodes affected by a change in the neighborhood topology (i.e., “hearing” of a new node) need to run it. Each node u , upon discovery of a new *suspect* node v , searches for such paths using its k -hop neighborhood knowledge. While the algorithm may run for general k -hop detection, our simulation studies (presented in Section 5) showed that $k \leq 2$ is sufficient for various densities of a network.

Algorithm 1: The *Wormhole Detection* Algorithm

Data: *SuspectList*(u) and k -hop Neighborhood Information of u where $k = 1, 2$.

Result: For each node v_i in *SuspectList*(u), compute whether v_i is legitimate or not.

```

begin
  for every  $v_i$  in SuspectList( $u$ ) do
     $v_i \rightarrow$  NOT legitimate;
    /** Look for small path between [ $u, v_i$ ] */
    if [ImmediatePath( $u, v_i$ )]  $\vee$  [1-HopPath( $u, v_i$ )]  $\vee$  [2-HopPath( $u, v_i$ )]
    then
      /** A path has been found */
       $v_i \rightarrow$  legitimate;
    end
    continue with next node  $v_i$  in SuspectList( $u$ );
  end
end

```

The code of WDA is given in Algorithm 1. We will refer to the set of *suspected nodes* of each node u , as $SuspectList(u)$. A node v is labeled as suspect when it is the first time we hear from it and we want to perform the local path existence test described in Section 4.1. As we mentioned above, the output of this test indicates whether this suspect node is *legitimate* (and can be safely added to the neighborhood list of u) or might be the result of a wormhole in progress.

Each node u maintains the list of 2-hop neighbors $N^2(u)$. WDA performs the path existence test for all nodes $v \in SuspectList(u)$. The test consists of three steps, each of which is activated upon *failure* of the previous one.

ImmediatePath(u, v): This step tries to identify a *direct path* between the testing pair $[u, v]$. It works by having the active node u check if at least one of v 's neighbors is included in its neighborhood list. This is possible since u knows the 1-hop neighbors of v . In order to perform this check, u traverses both its own and v 's neighborhood lists. Thus, the time needed is $O(d^2)$, where d is the average density of the network.

1-HopPath(u, v): Active node u checks whether one of its 1-hop neighbors is *directly* connected to one of the 1-hop neighbors of v . This requires node u to traverse the neighborhood lists of all its 1-hop neighbors and check if at least one of v 's neighbors is included. This takes time $O(d^3)$, where d is the average density of the network. Intuitively, this is done by considering whether u and v have two neighbors x and y , respectively, that are *directly* connected.

2-HopPath(u, v): In this last step node u checks whether one of its 1-hop neighbors shares a common neighbor with one of the 1-hop neighbors of v . Thus, the time needed is of order $O(d^4)$. This is done by considering whether u and v have two neighbors x and y , respectively, that have a common 1-hop neighbor z . This alternative path, if it exists, will be an attestation that v is a legitimate node. Otherwise, node u can conclude with very high probability that "hearing" v is the result of a wormhole, without the need of checking the existence of k -hop paths for $k > 2$.

5 Simulation Results

In this section, we present the practical impacts of the proposed wormhole detection algorithm. The experiments were deployed both in a simulator and a real sensor environment. Two properties were of special interest: *i*) Path Existence Percentage, and *ii*) Detection Time. Since the algorithm always prevents a wormhole attack, our focus is mostly on legitimate node addition.

5.1 Performance Evaluation

In order to evaluate our wormhole detection algorithm, we tested its success in finding small paths between a pair of nodes $[u, v]$. We generated random network topologies by placing 500 nodes uniformly at random with an average density d varying between 4 and 15. To ensure statistical validity, we repeated

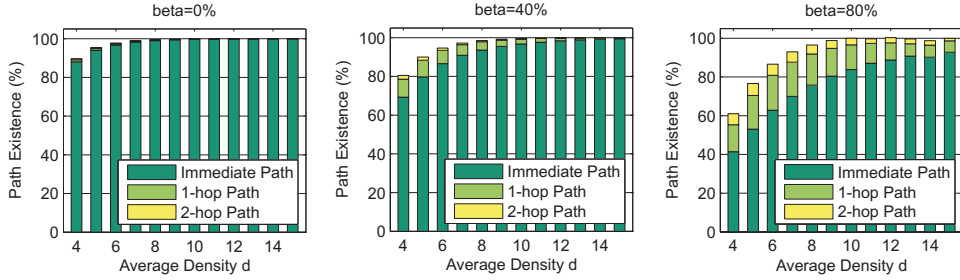


Fig. 3. Path Existence percentage between a pair of nodes $[u, v]$ when a fraction, β , of u 's neighbors are excluded from the WD algorithm. (a) $\beta = 0\%$ (b) $\beta = 40\%$ (c) $\beta = 80\%$

each experiment 1000 times and averaged the results. Once a topology was created, we started randomly adding new nodes in the network. This triggered the surrounding nodes to run WDA. In order to achieve real scenario simulations, we had to take into consideration possible missed route updates or incomplete neighborhood information. Thus, we used a variable β which indicated the percentage of a node's neighbors that were excluded during the path existence test. In other words, β is a fraction of u 's neighbors that are not taken into account when u is trying to find a *small* path with v . These “excluded nodes” were selected at random from u 's neighborhood, for increasing values of β .

Figure 3 depicts the path existence percentage between a pair of nodes $[u, v]$. This was broken down to the existence of an *immediate* path (1st step of WDA), *1-hop* path (2nd step), and *2-hop* path (3rd step). As we can see, our algorithm provides very good results even for low densities and when a large number of u 's neighbors is excluded. In general, the following observations can be made by the simulation results:

- WDA provides very good results (almost 100% success in finding a small path) despite the network density (Figure 3(a)).
- The more expensive 1-hop and 2-hop path tests are rarely need to be executed since in most of the cases the immediate path test is sufficient. This results in faster detection. Also, it loosens the need of the 2-hop neighborhood maintenance by a node (Assumption SMA-1).
- Existence probability does drop for large β , but only for low density cases. However, in such cases, the usefulness of the network also drops. Even when $\beta = 80\%$ (Figure 3(c)) the existence percentage is almost 100% for densities $d \geq 7$. Thus, one may reside only in the 1-hop test to increase the success probability. The usefulness of the 2-hop test seems to be questionable since even for large β (80%) its contribution remains small. This suggests that the 2-hop test can be dropped entirely.

In summary, our algorithm allows legitimate nodes to be added with high probability even for low density networks. In such networks, one can adjust WDA to just consider 1-hop neighborhoods and perform only the first check (for more, see also Section 6). Furthermore, since the algorithm does not progress to the next test unless the previous one has failed, in more than 95% of the cases WDA will conclude after the first check, keeping the overall test time relatively small.

5.2 Implementation and Experiments on real sensor devices

In this section, we present some results on the detection time of the algorithm. These are collected through experiments from our implementation of WDA on real sensor devices³. The goal is to justify the practicality of our approach from an implementation and real deployment point of view.

Table 1. Size of the compiled code, in bytes

Module	RAM usage	Code Size
Neighborhood Discovery	136	968
WDA	104	766
Total	240	1734

We start with the *memory footprint*, an important measure of WDA’s feasibility and usefulness on memory constrained sensor nodes. Table 1 lists the memory requirements of the necessary modules. The Neighborhood Discovery module is the one responsible for creating a node’s k -hop neighborhood ($k = 1, 2$). The WDA module contains all the necessary methods described in Section 4.3. As we can see, in total, the algorithm consumes 240 bytes of RAM and 1734 bytes of code memory. This leaves enough space in the mote’s memory for user applications and other protocols. For example, the total RAM available in Telos motes is 10 KB.

Next we measured the time each test of WDA required. Figure 4(a) shows the measured mean times for each of the three tests, for different network densities. The immediate and 1-hop path tests, which cover more than 95% of the cases, conclude in much less than 1 second (around 100ms and 550ms, respectively). The most time consuming step is the 2-hop test. However, as we mentioned before, it will rarely need to be executed and it can even be dropped entirely.

We have to note that this experiment was conducted with nodes maintaining their neighbors in a typical list that is not sorted or pre-processed in any way. However, as Figure 4(b) shows, having neighbor lists sorted by node IDs significantly decreases the detection time of WDA’s steps, and in particular that of the 2-hop test. In any case, the time of WDA is very small fulfilling the need of

³ The current development of WDA builds on Moteiv Telos motes - a popular architecture in the sensor network research community.

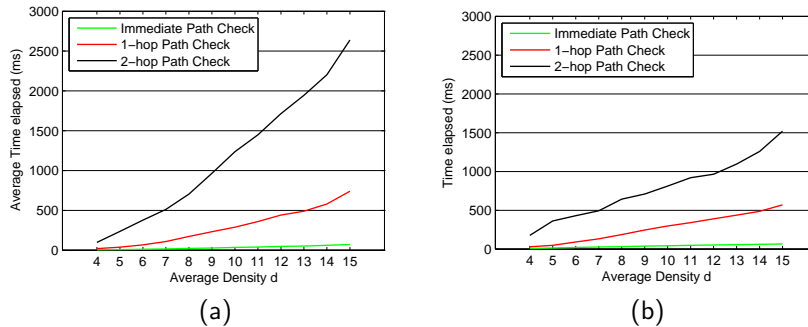


Fig. 4. Running time of WDA for different network densities. (a) Node IDs are randomly placed in a neighborhood list (b) Neighborhood list is kept sorted

immediate response in case of a wormhole in progress. Dropping the 2-hop test entirely does not significantly affect the success probability while at the same time it decreases the running time considerably. In such case there is no need to resort in sorted neighborhoods or other data structures to speedup computation, however, we leave this decision open to the particular implementation.

6 Discussion and Critique

In this work we proposed a method that identifies and prevents wormhole attacks. Key to the method is the observation that in the case of an attack, no real path of just a few hops will ever exist between the wormhole endpoints (Section 3.2, Assumption AMA-1) since otherwise it would not be possible to distinguish between real and fake short paths. It is, however, a very realistic assumption since the point of the attack is to make a distant node appear closer to a point of interest (say the base station) and attract as much traffic as possible [1, 13]. Decreasing the radius of the wormhole, will also decrease the effect of the attack on the network in terms of the numbers of sensors that use the wormhole link [2].

Of course an attacker can still try to fool the algorithm by establishing many smaller wormholes that are connected in a series but this has two shortcomings. First, it takes a lot of time, effort and hardware which will increase the risks of detection. Second, it can be defeated easily by our method if we just consider 1-hop neighborhoods ($k = 1$). In this case no wormhole link of length bigger than 2 can ever be established while, as demonstrated in Figure 3, addition of legitimate nodes can occur with high probability even for small density networks.

For the same reason, assumption SMA-1 may also be weakened by maintaining only 1-hop neighborhoods among nodes, a typical procedure for any routing protocol. In practice well connected networks have node densities bigger than 7 or 8 in which case the overhead of maintaining 2-hop neighborhoods does not offer any significant advantage over the 1-hop case (Figure 3). Our only important assumption is that there is some initial interval where the nodes have

safely established their neighborhood information (Assumption AMA-2). This is a standard assumption made in many of the works in this area [2, 13, 14, 17–19] but also in works where a security infrastructure has to be established.

We should mention that one way our method could be defeated is if the attacker has compromised a few nodes in the neighborhood of the wormhole endpoints in which case the compromised nodes may “lie” about their true neighbors. This combined attack, however, is *not* a wormhole attack anymore. *All* cited results consider the attack in which an adversary *simply* forwards messages from one part of the network to another. This combined attack, however, is an interesting future direction that we are planning to pursue further.

Finally, we should emphasize that contrary to prior work [2, 14], we don’t assume that the network is *static*. In fact a large portion of this work considers *dynamic* changes in the neighborhood structure of nodes.

7 Conclusions

A wormhole attack is considered to be a prominent attack that is carried out in order to alter the correct functioning of a wireless network. The detection of such an attack is still a significantly challenging task. In this paper, we have studied the problem of wormhole detection in sensor networks. We have provided a snapshot of the current state of the art, discussed existing solutions and listed their behavior and limitations. More importantly, however, we have proposed a practical detection algorithm based on the observation that no real path of just a few hops will ever exist between the wormhole endpoints.

We investigated the effectiveness of our proposed algorithm both analytically and experimentally. Our results have confirmed that it can always prevent a wormhole attack, while at the same time it allows legitimate node addition with high probability, even for low density networks. Furthermore, the implementation details show that it is lightweight enough to run on sensor nodes in terms of both memory requirements and processing overhead. In general, we believe that this algorithm will have a practical use in real-world deployments and can be considered as a reference point for further investigation of more attractive solutions against wormhole attacks since it does not require any specialized hardware, tight clock synchronization, or distance measurements between the nodes.

References

1. N. Ahmed, S. S. Kanhere, and S. Jha. The holes problem in wireless sensor networks: a survey. *Mobile Computing and Communications Review*, 9(2):4–18, 2005.
2. L. Buttyán, L. Dóra, and I. Vajda. Statistical wormhole detection in sensor networks. pages 128–141. 2005.
3. J. Eriksson, S. V. Krishnamurthy, and M. Faloutsos. Truelink: A practical countermeasure to the wormhole attack in wireless networks. pages 75–84, 2006.

4. T. H. Hai and E.-N. Huh. Detecting selective forwarding attacks in wireless sensor networks using two-hops neighbor knowledge. In *NCA '08: Proceedings of the 2008 Seventh IEEE International Symposium on Network Computing and Applications*, pages 325–331, Washington, DC, USA, 2008. IEEE Computer Society.
5. Y.-C. Hu, A. Perrig, and D. B. Johnson. Wormhole attacks in wireless networks. *IEEE Journal on Selected Areas in Communications*, 24(2):370–380, 2006.
6. N. B. S. Issa Khalil, Saurabh Bagchi. Liteworp: A lightweight countermeasure for the wormhole attack in multihop wireless networks. page 22, 06 2005.
7. C. Karlof and D. Wagner. Secure routing in wireless sensor networks: Attacks and countermeasures. *AdHoc Networks Journal*, 1(2-3):293–315, September 2003.
8. T. Korkmaz. Verifying physical presence of neighbors against replay-based attacks in wireless ad hoc networks. In *ITCC '05: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume II*, pages 704–709, Washington, DC, USA, 2005. IEEE Computer Society.
9. I. Krontiris, Z. Benenson, T. Giannetsos, F. Freiling, and T. Dimitriou. Cooperative intrusion detection in wsn. In *6th European Conference on Wireless Sensor Networks (EWSN 09)*, Cork, Ireland, February 2009.
10. I. Krontiris, T. Giannetsos, and T. Dimitriou. Launching a sinkhole attack in wireless sensor networks; the intruder side. In *SecPriWiMob '08: First International Workshop on Security and Privacy in Wireless and Mobile Computing, Networking and Communications*, Avignon, France, October 12-14.
11. M. Lehsaini, H. Guyennet, and M. Feham. a-coverage scheme for wireless sensor networks. In *ICWMC '08: Proceedings of the 2008 The Fourth International Conference on Wireless and Mobile Communications*, pages 91–96, Washington, DC, USA, 2008. IEEE Computer Society.
12. M. Lehsaini, H. Guyennet, and M. Feham. CES: Cluster-based energy-efficient scheme for mobile wireless sensor networks. In *IFIP Conference on Wireless Sensor and Actor Networks*, Ontario, Canada, July 2008.
13. R. Maheshwari, J. Gao, and S. R. Das. Detecting wormhole attacks in wireless networks using connectivity information. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 107–115, 2007.
14. R. Poovendran and L. Lazos. A graph theoretic framework for preventing the wormhole attack in wireless ad hoc networks. *Wireless Networks*, V13(1):27–59, January 2007.
15. S. S. Capkun, L. Buttyan, and J.-P. Hubaux. Sector: secure tracking of node encounters in multi-hop wireless networks. In *SASN '03: Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, pages 21–32, New York, NY, USA, 2003. ACM Press.
16. A. Savvides, C.-C. Han, and M. B. Strivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 166–179, New York, NY, USA, 2001. ACM Press.
17. B. B. W Wang. Visualization of wormholes in sensor networks. 10 2004. in Proceedings of ACM Workshop on Wireless Security (WiSe), in conjunction with MobiCom.
18. W. Wang, B. Bhargava, Y. Lu, and X. Wu. Defending against wormhole attacks in mobile ad hoc networks: Research articles. *Wirel. Commun. Mob. Comput.*, 6(4):483–503, June 2006.
19. W. Wang and A. Lu. Interactive wormhole detection and evaluation. *Information Visualization*, 6(1):3–17, 2007.