2007

Securing Virtual Coordinate System Based Routing in Wireless Sensor Networks

Jing Dong

Brett Bavar

Cristina Nita-Rotaru
Purdue University, crisn@cs.purdue.edu

Report Number:
07-009

SECURING VIRTUAL COORDINATE SYSTEM BASED ROUTING IN WIRELESS SENSOR NETWORKS

Jing Dong
Brett Bavar
Cristina Nita-Rotaru

Department of Computer Science
Purdue University
West Lafayette, IN 47907

CSD TR #07-009
April 2007

Securing Virtual Coordinate System Based Routing in Wireless Sensor Networks

Jing Dong Brett Bavar Cristina Nita-Rotaru
Department of Computer Science, Purdue University
305 N. University St., West Lafayette, IN 47907 USA
{dongj,bbavar,crisn}@cs.purdue.edu

Abstract— **Virtual coordinate system (VCS) based routing provides a practical, efficient and scalable means for point-to-point routing in wireless sensor networks. Several VCS-based routing protocols have been proposed in the last few years, all assuming that nodes are cooperative. However, malicious nodes may violate this assumption, making VCS-based routing protocols vulnerable to numerous attacks. Thus, it is critical to provide security mechanisms for these protocols to ensure correct operations in adversarial deployment environments.**

In this work, we study the security of VCS-based routing protocols. We identify new attacks targeting the accuracy and stability of virtual coordinates that VCS-based routing relies on and propose several defense mechanisms against the identified attacks. We evaluate the impact of the attacks and the effectiveness of our defense mechanisms using a well-known VCS-based routing protocol, BVR.

I. Introduction

Wireless sensor network designs have evolved in recent years, from primarily focusing on data collection [1] to more sophisticated tasks such as data centric storage [2], [3]. Likewise, the requirements for communication protocols have also evolved, from the basic many-to-one and one-to-many communications to more sophisticated point-to-point communications. Well-known wireless routing protocols such as AODV [4] and DSR [5] are not appropriate solutions for sensor networks as they do not scale well for large networks and have relatively high overhead. Virtual coordinate system (VCS) based routing protocols have been proposed to overcome these limitations. In such protocols, routing is performed in a greedy manner based on virtual (or logical) coordinates obtained through a virtual coordinate establishment mechanism integrated with the routing protocol or through an external virtual coordinate system. VCS-based routing protocols require only local interactions and minimal state information that does not grow with the size of the network. As a result, such protocols have increased scalability and reduced overhead.

Although several VCS-based routing protocols have been proposed in the last few years [6], [7], [8], [9], there has been little work that investigates the security of such protocols. As many applications for wireless sensor networks require deployment in adversarial environments, it is critical to provide security mechanisms to make these protocols operate correctly in the presence of attackers.

In this paper, we study the problem of securing VCS-routing protocols. Like other routing protocols, VCS-based routing protocols are vulnerable to numerous attacks that seek to disturb the routing service by injecting, modifying, replaying or dropping packets. In addition, VCS-based routing protocols depend significantly on the accuracy and stability of the virtual coordinates. Our contributions are:

- We identify and analyze new attacks against VCS-based routing protocols that target the virtual coordinates. We refer to the attacks as *coordinate inflation, deflation, oscillation, and pollution* attacks. We show that virtual coordinate accuracy can be significantly influenced by *coordinate inflation, deflation* and *pollution* attacks while virtual coordinate stability can be influenced by *coordinate oscillation* attacks.
- We evaluate experimentally the impact of the attacks on the accuracy and stability of virtual coordinates and on the routing performance, using a well-known VCS-based routing protocol, BVR [8]. Our experiments show that coordinate deflation is the most damaging attack, while coordinate inflation creates the least damage.
- We propose defense mechanisms against virtual coordinate attacks. Specifically, we use hop-by-hop authentication and a novel wormhole detection technique to address coordinate deflation. We introduce a coordinate variance based parent selection mechanism to mitigate coordinate oscillation attacks and a coordinate replication technique to mitigate coordinate pollution attacks.
- We demonstrate the effectiveness of the proposed defense and mitigation mechanisms through simulations using an implementation of the BVR protocol in the TOSSIM [10] simulator.

The rest of the paper is organized as follows. Section II provides an overview of the main mechanisms involved in the design of VCS-based routing protocols. Section III presents attacks against the virtual coordinates. Section IV outlines several defense mechanisms. Section V demonstrates the impact of the attacks and the effectiveness of our defense mechanisms using the BVR

routing protocol. Section VI overviews related work and Section VII concludes our paper.

II. Overview of VCS-based Routing Protocols

In this section, we provide an overview of the common design of VCS-based routing protocols.

VCS-based routing protocols are geographical routing protocols that forward packets to the neighbor that is closest to the destination. Instead of using physical coordinates, VCS-based routing protocols use virtual or logical coordinates obtained through an integrated virtual coordinate establishment mechanism or a virtual coordinate system external to the routing protocol. In the following, we assume the virtual coordinate system is integrated with the routing protocol.

Most VCS-based routing protocol designs share four major components: (1) virtual coordinate establishment, (2) destination node coordinate lookup, (3) greedy routing, and (4) a fall-back or complementary procedure. Virtual coordinate establishment is achieved based on a set of reference nodes that can be special infrastructure nodes, such as landmarks [6], or regular sensor nodes [7], [8]. The virtual coordinates are established based on the connectivity graph of the network. A common approach is for the reference nodes to periodically broadcast coordinate messages in the network. The flooding of these messages across the network builds the shortest path trees rooted at each of the reference nodes among all the nodes in the network. For convenience, the parent of a node on the shortest path tree is referred to as the parent of the node. The hop count from a reference node accumulates on the packet as the packet is flooded across the network. The network coordinates of a node are then derived based on the hop counts to each of the reference nodes.

In order to route a message to a destination the source node must be able to lookup the coordinates of the destination node. Many protocols use a set of coordinate servers to maintain coordinates for the nodes in the network. Nodes are mapped to a coordinate server by using a hash function. A node is responsible for informing the coordinate server storing its coordinates of any change. In general, the number of changes is low in static wireless sensor networks.

Once a node obtains the coordinates of the destination node, VCS-based routing follows the geographic routing paradigm, in which each node forwards the message to the neighbor that is the closest to the destination under some protocol specific distance metric.

Finally, if the greedy routing reaches a node that is closer to the destination than all of its neighbors (i.e. a local minima), a protocol specific fall-back procedure is invoked. For example, in the case of the protocol in [8], the fall-back procedure re-directs the message to the reference node (known as a beacon) closest to the destination. When the message reaches that beacon node, it is then flooded in the network. Typically, the fall-back procedure incurs much more overhead than the greedy forwarding process.

Two properties of the virtual coordinate establishment that influence the performance of VCS-based routing protocols are *accuracy* and *stability*. Accuracy captures the difference between the perceived coordinates of the nodes and their actual coordinates, while stability captures the frequency of coordinate changes. Inaccurate coordinates may cause messages to be routed in a wrong direction; unstable coordinates, on the other hand, can cause route flapping and incur additional coordinate maintenance overhead.

III. Attacks Against VCS-based Routing

In this section we present several new attacks against VCS-based routing protocols. The attacks are specific to VCS-based routing protocols and exploit the fact that such protocols depend heavily on the accuracy and stability of the virtual coordinates. After describing the assumptions we make about adversaries, we outline some general attacks against wireless sensor networks. We then describe attacks against the virtual coordinate establishment and the coordinate lookup component of VCS-based routing protocols. Finally, we exemplify how the newly identified virtual coordinate attacks can be exploited to attack the greedy-routing protocol.

A. Adversarial Model

We assume that the radio links are insecure. The attackers can eavesdrop on the radio transmissions, inject packets, modify packets, and replay previously overheard packets. We assume "mote" class attackers [11], that is, the attacker nodes are similar in capabilities as legitimate nodes. We also do not assume the legitimate sensor nodes are tamper resistant. The attacker may compromise some legitimate nodes and get full control on the operations of the compromised nodes. She can also extract all the secret keys and any other data in the compromised nodes to share with other attacker nodes. The attacker may also arrange multiple attacker nodes to collude via high quality communication links and establish wormholes.

The physical and MAC layers are susceptible to direct attacks. The attacker can jam the network with high power radio transmitters. She can also more stealthily mount DoS attacks on the MAC by exploiting the specific vulnerabilities of the MAC protocol. The physical layer DoS attacks are typically countered by frequency hopping or spread spectrum techniques. The MAC layer attacks can be addressed with more secure MAC protocols, such as [12]. While these attacks are real and

dangerous, we consider them out of the scope of the paper.

We do not focus on general attacks against sensor networks, such as Sybil attacks [13], or node replication attacks [14], in which a single adversary can control a significant fraction of the network by claiming multiple identities or cloning a subset of physical devices, respectively. We assume that techniques such as [15] are employed to address Sybil attacks and [14] to address node replication attacks.

B. General Attacks on Wireless Sensor Networks

Several basic attacks against sensor networks were outlined in [11]. Some of these attacks can be used in the virtual coordinate attacks we identify. To facilitate the presentation of the virtual coordinate attacks, we summarize them here:

Spoofing, altering, or replaying packets: The attacker exploits the open nature of wireless communication and the lack of authentication to spoof, alter, or replay control or data packets. Such attacks can usually be prevented using authentication protocols and freshness mechanisms such as timestamps, nonces, and sequence numbers.

Selective forwarding: The attacker does not correctly provide the routing service by selectively dropping some of the packets. This type of attack requires the attacker to be an insider and cannot be addressed by means of authentication.

Wormhole: One attacker node forwards the overheard messages via some out-of-band channel to another attacker node in a distant network region where the messages are replayed. The wormhole attack generally causes confusion on the neighborhood relationship among legitimate nodes, which usually damages normal operations of the upper layer protocols. Previously proposed solutions to address this problem require tight clock synchronization and topology information [17] not necessarily available in sensor networks, specialized hardware such as directional antennas [18], or are designed for specific MAC protocols [19] not appropriate for sensor networks.

C. Virtual Coordinate Attacks

VCS-based routing protocols rely significantly on the accuracy and stability of the virtual coordinates. A class of attacks specific to VCS-based routing protocols target the virtual coordinate establishment and coordinate lookup components, causing the resulting coordinates to be unusable to the routing protocol. Specifically, the attacker aims to cause legitimate nodes to have incorrect coordinates, unstable coordinates, or both. Now we classify the attacks against virtual coordinates based on their intended effect on the coordinate system established.

1) Coordinate Deflation Attack: The goal of this attack is to cause legitimate nodes to obtain network coordinates smaller than their actual coordinates. One way an attacker can mount this attack is by announcing incorrect small coordinates in its neighborhood. The attacker nodes can be either compromised legitimate nodes or outside malicious nodes spoofing legitimate nodes. Another way the attacker can mount this attack is by using the wormhole attack. If one attacker node is located close to a reference node, it can tunnel the overheard legitimate announcements with small coordinates to another attacker node in a distant network region. By replaying these tunneled coordinate announcements, the attacker can cause legitimate nodes in the distant region to derive incorrect small coordinates.

2) Coordinate Inflation Attack: The goal of this attack is to cause legitimate nodes to obtain larger coordinates than their actual coordinates. Similar to the coordinate deflation attack, the attacker can announce artificially enlarged coordinates by either compromising legitimate nodes, spoofing legitimate nodes, or tunnelling and replaying legitimate announcements of large coordinates from a distant network region.

Unlike the coordinate deflation attack, the coordinate inflation attack is not always effective. If the legitimate nodes have a path to the reference node that does not pass through any attacker nodes, the legitimate nodes can still derive correct network coordinates from the coordinate message forwarded on the correct path. However, if the attacker nodes form a vertex-cut between the legitimate nodes and the reference nodes, the incorrect large coordinate announcements from the attacker nodes directly inflate the coordinates of these legitimate nodes. If the protocol considers path quality when determining coordinates, the coordinate inflation attack can also be made more effective by having the attacker nodes announce large coordinates but with good path quality.

3) Coordinate Oscillation Attack: The goal of this attack is to cause instability of the virtual coordinates. The attacker can mount this attack by alternatively making announcements with small coordinates and large coordinates or by making random coordinate announcements. As a result, the legitimate nodes near the attacker node adjust their coordinates accordingly and oscillate between large and small coordinates. Similar to the coordinate inflation and deflation attacks, the fake coordinate announcements made by the attacker nodes can either be from compromised nodes, spoofing nodes, or be wormhole tunneled from other regions of the network.

4) Coordinate Pollution Attack: The operation of the VCS-based routing relies on the correct information of the destination coordinates. Unlike coordinate inflation, deflation, or oscillation attacks which cause nodes to incorrectly compute their coordinates, the goal of a

coordinate pollution attack is to influence the coordinate lookup so that source nodes obtain incorrect coordinates of the destination nodes they need to route to.

We discuss two approaches that the attacker can use to mount this attack. As described in Section II, virtual coordinate systems use coordinate servers to store the virtual coordinates of all the nodes in the network and to respond to coordinate queries from possible source nodes looking up coordinates of destination nodes. If the attacker can compromise one or more coordinate servers, she can give arbitrary responses to the coordinate queries directed to adversarial controlled servers.

Alternatively, the attacker can mount the attack in a way similar to the rushing attack [20]. If the attacker can spoof the coordinate servers, she can directly forge responses to position queries and send the forged response to the victim node before the response from the actual coordinate server arrives. If the victim node uses only the first query response and ignores others, it will use the wrong destination coordinate information for all the messages destined to the target node.

The danger of the coordinate pollution attack lies in the fact that the affected messages are usually forwarded futilely over many hops toward wrong or non-existent positions in the network. Eventually, the messages are either dropped when the TTL expires or the expensive fall-back mode is invoked on the message if the message reaches a local minimal before the TTL expires. Therefore, incorrect destination coordinates not only cause a significant number of routing failures, but also incur a significant amount of resource consumption in the network.

D. Impact of Virtual Coordinate Attacks

An attacker can disrupt normal message routing in the network by directly disrupting the virtual coordinate establishment and the node coordinate lookup process. In addition, since the success of the greedy routing is the key to the low overhead of a typical VCS-based routing protocol, the attacker can also indirectly impede the correct functioning of the network by causing a large number of greedy routing failures. The frequent invocation of the expensive fall-back mode in the routing protocol causes an overwhelmingly high overhead in the network and degrades the routing performance.

The vulnerabilities on the virtual coordinates may also be exploited to magnify the attacker's power in attacking some specific VCS-based routing protocol. For example, in BVR, nodes prefer neighbors with small coordinates as next hop for message forwarding. By using the coordinate deflation attack, the attacker node artificially deflates the network coordinates in the region thus attracting a significant portion of the routing traffic in the network. Once the traffic is diverted through the attacker node, it can either behave as a gray-hole by selectively dropping routing traffic to mount more targeted attacks or behave as a black-hole by dropping all routing traffic.

We note that in the context of wireless sensor networks, as nodes rely on other nodes to derive their virtual coordinates, all of the virtual coordinate attacks are "contagious," in the sense that legitimate nodes once affected by the attack become "attackers" themselves, and propagate the effect of the attack further in the network. For example, for the coordinate deflation attack, once the direct neighbors of the attacker node derive incorrect small coordinates for themselves, they announce these incorrect coordinates in their neighborhoods and cause their neighbors to have incorrect small coordinates as well. This infection process continues throughout the whole network until the incorrect coordinate announcements reach the legitimate nodes that are closer to the actual reference node than to the attacker node. A back-of-envelope calculation reveals that a well positioned attacker can result in as much as 80% of the nodes in the network to have wrong coordinates. Similarly, the effect of a coordinate inflation attack and a coordinate oscillation attack can be propagated throughout a large portion of the network with a single well positioned attacker node. Due to the potential network wide impact that can be achieved by a small number of attackers, these attacks are particularly dangerous.

IV. Mitigating Virtual Coordinate Attacks

In this section, we describe the mechanisms to mitigate the virtual coordinate attacks described in the previous section. We first describe our assumptions, then we focus on the defense against the attacks. Our goal is to maintain accurate and stable virtual coordinates and high routing performance with minimum overhead despite the presence of attackers. To avoid *blacklist* attacks where a node spreads false rumors about other nodes, we adopt the principle that a node makes decision only based on their own observations.

A. Assumptions

We assume broadcast authentication is available, which can be achieved by using existing techniques such as μTESLA [21]. We also assume the existence of symmetric keys between any two nodes in the network, which can be achieved by using a pre-distributed key management scheme for sensor networks such as [22], [23], [24], [25], [26]. This enables data source authentication, excluding the possibility of outsider attacks, such as message spoofing, altering, and injection.

We assume the existence of a secure neighbor verification method that allows each node to verify direct neighbor relationships. Since such a verification

method is not invoked unless the presence of attacks is positively detected, simple and inexpensive mechanisms are sufficient. For example, one may implement such a verification method by measuring and comparing the RTT to the claimed neighbor with the estimated one-hop RTT. Finally, we assume that the network is static and relatively stable, and that there is a period of time, such as the initial period after the network deployment, when the network is not under attack.

B. Mitigating Inflation and Deflation Attacks

An attacker mounts the inflation and deflation attacks by making malicious coordinate announcements or by tunneling overheard legitimate coordinate announcements from one network region to another network region. Data source authentication prevents attacker nodes from spoofing or injecting packets. However, the correct derivation of the coordinates of each node relies on other nodes in the network correctly updating the coordinate messages as they are flooded across the network. Thus, further mechanisms are necessary to prevent attacker nodes from making malicious changes to the coordinate messages. In the following, we first describe a hash chain based technique that prevents attacks in the absence of wormholes, then we address the wormhole based attacks.

Our techniques can be adapted to work for both inflation and deflation attacks. However, deploying the protection mechanisms for both attacks will double the overhead of the protocol. As noted in the previous section and validated in the experiments (Section V-B), the coordinate inflation attack has only limited impact in general network topologies. Thus, we only focus on the coordinate deflation attack.

1) Preventing Deflation Attacks Based on Malicious Coordinate Message Updates: Most VCS-based routing protocols use a hop count and/or some hop count like field (e.g. path quality) in the coordinate messages for deriving network coordinates. In the following, we discuss a hash chain based technique similar to the one in [27] for protecting the hop count field in detail. We note that this method can be easily adapted to protect other hop count like fields that are incremented at each hop.

To use the hash chain to protect the hop count field, the reference node first generates a random number r and uses a one-way hash function H to generate a hash chain,

$$r, H(r), H^{(2)}(r), \ldots, H^{(N)}(r),$$

where $H^{(i)}(r)$ denotes applying H on r iteratively for i times, and N is the estimated upper bound for the network diameter. The reference node then uses authenticated broadcast to disseminate the tuple $(H^{(N)}(r), N)$, referred to as the *anchor tuple*, throughout the network. When broadcasting coordinate messages, instead of including the plain hop count, the reference node includes the tuple $(0, r)$, referred to as the *hop count tuple*, in the message.

When a node receives the hop count tuple (i, v_i) in the coordinate message, it first verifies that $H^{(N-i)}(v_i) = H^{(N)}(r)$. If the verification is successful, the node determines its hop count as $i+1$ and forwards the tuple $(i+1, H(v_i))$ to their neighbors.

Note that with the above approach, it is impossible for a node to claim a hop count that is smaller than its actual hop count by more than one, unless they are able to invert the hash function H. The attacker node, however, can claim one less hop count by replaying the hop count tuple it received from the previous hop. Such attacks only result in a decrease of one in the coordinates of the attacker node, and we expect them to have only limited impact in affecting the coordinates of other nodes.

2) Detection of Deflation Attacks Based on Wormholes: Besides making malicious updates to the coordinate messages directly, wormholes present another viable means for mounting the deflation attacks. We now present a technique for detecting and eliminating wormhole based deflation attacks.

Our wormhole detection algorithm relies on the observation that in the presence of the coordinate deflation attack the hop count distribution of the nodes differs significantly from the distribution in the case without attacks. More specifically, the coordinate deflation attack causes more nodes to have small hop counts and fewer nodes to have large hop counts. In a naive approach, we can detect such changes, hence the presence of the attack, by querying the coordinates of all the nodes in the network. In the following, we propose a more efficient algorithm based on statistical sampling for detecting such changes.

Our algorithm runs on a central entity (CE), which can be a base station or a laptop that sweeps through the network periodically. In the high level, the CE first estimates the hop count distribution, referred to as the reference distribution, for the case when the network is known to be not under attack (e.g. immediately after the network deployment). Then the CE periodically compares the estimate of the current hop count distribution with the estimated reference distribution using Pearson's χ^2-test [28] to determine if a change in the distribution has occurred. Now we describe the details of the method.

Characterizing the hop count distribution To efficiently estimate the hop count distribution, the CE performs simple random sampling (SRS) on the network by uniformly randomly selecting m nodes *with replacement*. Then the CE queries the sampled nodes for their coordinates and records (n_1, n_2), where n_1 is the number of nodes with hop count less than or equal to a hop count threshold k and n_2 is the number of nodes with hop count greater than k. The hop count threshold k is an

input to the algorithm. Optimally, k is selected such that in the presence of the attack, there are more nodes for all the hop counts less than or equal k and fewer nodes for all the hop counts greater than k. The optimal value for k maximizes the changes in n_1 and n_2 before and after the attack, thus maximizing the probability of detecting the changes. Typically, half of the network diameter is a good estimation for k. Note that since we sample with replacement, the unique number of nodes in the sample is most likely smaller than the sample size.[1]

Statistical change detection Let (r_1, r_2) and (s_1, s_2) be the numbers that the CE recorded in the time assumed with no attacks and in some other time for which we wish to check for the presence of attacks. We now perform the Pearson's χ^2-test as follows.

Let $E_1 = \frac{1}{2}(r_1 + s_1)$, $E_2 = \frac{1}{2}(r_2 + s_2)$, and

$$X^2 = \sum_{j=1,2} \frac{(r_j - E_j)^2 + (s_j - E_j)^2}{E_j},$$

then assuming the network is not under attack, we have X^2 follows the χ^2 distribution with 1 degree of freedom, and the P-value $p = Pr(\chi^2 > X^2)$ can be obtained with the standard statistical software. Let f (e.g. 0.05) be the acceptable false positive rate, we declare that the network is under attack if $p < f$.

Note that the above method assumes the hop count distribution in the network is static. To account for normal network varations, we can adjust (r_1, r_2) by a small amount δ and perform the above test only if $s_1 > r_1 + \delta$. The value of δ depends on the expected network variations and is empirically estimated.

Once a wormhole deflation attack is detected, the CE floods the network with a wormhole warning message. Upon receiving the warning message, each node performs neighbor verification on their parent node. If the verification fails, it regards the node as an attacker and re-calculates its coordinates based on its other neighbors.

C. Mitigating Coordinate Oscillation Attack

In the coordinate oscillation attack, honest nodes update their coordinates frequently due to the rapid changes in the coordinate announcements made by their parent node, which can be an attacker node or some other honest node affected by the attack. To mitigate this attack, we propose a stability based parent selection policy, under which more stable neighbors are preferred as the parent. We now present the details as follows.

To determine the stability of its neighbors, each node stores the last t coordinate announcements from each of its neighbors. The node then calculates the coordinate variance for each of the neighbors as follows. Let $(c_1, c_2, \cdots, c_t)$ be one of the coordinate components of the last t coordinate announcements from neighbor n, then the coordinate variance for the neighbor n with respect to the coordinate component is

$$S_n = \frac{1}{t-1} \sum_{i=1}^{t} (c_i - \mu_n)^2,$$

where $\mu_n = \frac{1}{t} \sum_{i=1}^{t} c_i$. A node rejects a neighbor as its parent if its coordinate variance for any coordinate component is larger than a threshold. The value for the threshold is empirically estimated by considering normal network variations.

Although initially the attacker node can cause close honest nodes to vary their coordinates rapidly and also appear as attacker nodes to other nodes, after t coordinate announcements are collected from the attacker node, all of its honest neighbors will reject to use the attacker node as their parent due to the large variance computed for the attacker node. Thus, the attacker node is quickly isolated by its neighbors and the network re-stablizes. As a side benefit, this stability based parent selection policy may also result in more stable coordinate system for networks not under attack.

We note that a smarter attacker node may overcome this defense mechanism by varying its coordinate announcements in a smaller range or less frequently so that its computed variance is below the threshold. However, the impact of such an attack is significantly smaller than the full scale oscillation attacks.

D. Mitigating Coordinate Pollution Attack

In a coordinate pollution attack, source nodes obtain incorrect virtual coordinates for destination nodes. Since the authentication mechanisms described in Section IV-A prevent spoofed replies to coordinate queries, the main threat for coordinate pollution attacks comes from compromised coordinate servers. We propose to mitigate such threats by using redundant coordinate servers. More specifically, each node stores its coordinates in multiple coordinate servers, instead of in only one coordinate server. When querying the coordinates for a target node, the node sends the query to a random coordinate server from all of the coordinate servers for the target node. If the coordinates obtained from the selected coordinate server result in poor routing performance, the node queries all of the coordinate servers for the target node. Then through a majority voting scheme, the node determines which coordinate servers are malicious and refrains from using those servers for future queries.

We note that if the attacker compromised a majority of the coordinate servers maintaining the coordinates for some node, then there is no solution [29] to guaranteeing that coordinate lookups for that node will return correct results.

[1]The expected value for the unique number of nodes in a sample of size m from a total of n nodes is $n(1 - (1 - 1/n)^m)$

V. Experimental Results

In this section, we evaluate the impact of the attacks and the effectiveness of our proposed defense and mitigation mechanisms using an implementation of the well-known VCS-based routing protocol, BVR [8] in the TOSSIM [10] simulator for sensor networks. We selected BVR because it is a mature protocol which was shown to perform well in non-adversarial environments. In BVR, the reference nodes and coordinate messages used to compute coordinates are referred to as beacons and beacon messages, respectively. The beacons also serve as coordinate servers. The coordinates in BVR are described as a vector of hop-counts to each beacon. Below we will use this terminology to describe the experiments and discuss the results.

A. Experiment Setup and Metrics

The network consists of 100 nodes uniformly randomly distributed. The radio links are generated with the `LossyBuilder` tool included in TOSSIM, which generates probabilistic links based on empirical measurements from real motes. The average node degree is 12. We randomly select 8 nodes to be beacons, each of which floods a beacon message every 10 seconds. For evaluating the routing performance, we make a routing request between two randomly selected nodes every second. For evaluating the impact of the attacks, the attackers are randomly selected among the nodes and all attackers drop all data packets passing through them. The duration of each experiment is 2000 seconds. The experiment results are the average of 10 different runs with different random topologies.

We seek to evaluate the accuracy and stability of the virtual coordinates and their effect on the routing service. We use n to denote the total number of nodes, m to denote the number of beacons, i.e. the dimensions of the coordinates. The correct coordinates of node i are denoted as $\vec{c_i} = (c_{i1}, c_{i2}, \ldots, c_{im})$, while the perceived coordinates (due to attacks) of node i are denoted as $\vec{x_i} = (x_{i1}, x_{i2}, \ldots, x_{im})$.

Virtual coordinate accuracy: We characterize virtual coordinate accuracy by using the absolute errors in the coordinates for individual nodes and for the system as a whole. We define the error of the jth component of node i's coordinates, e_{ij}, as $e_{ij} = |x_{ij} - c_{ij}|$, and the error of the coordinates of node i as $e_i = \sum_{j=1}^{m} e_{ij}$. We use mean absolute error (MAE) to characterize system-wide errors. The system-wide error of the jth coordinate component E_j is defined as $E_j = \frac{1}{n}\sum_{i=1}^{n} e_{ij}$, and the system-wide error E is defined as $E = \sum_{j=1}^{m} E_j$.

Virtual coordinate stability: We characterize virtual coordinate stability by using the variance of the coordinates for individual nodes and for the system as a whole. We sample the coordinates of all nodes in the network periodically. Let T be the total number of samples taken and $\vec{x_i}^t = (x_{i1}^t, x_{i2}^t, \ldots, x_{im}^t)$ be the coordinates of node i at the tth sample. We define the instability value of the jth component of node i's coordinates s_{ij} as $s_{ij} = \frac{1}{T-1}\sum_{t=1}^{T}(x_{ij}^t - \bar{x_{ij}})^2$, where $\bar{x_{ij}} = \frac{1}{T}\sum_{t=1}^{T} x_{ij}^t$. The instability value for the coordinates of node i is defined as $s_i = \sum_{j=1}^{m} s_{ij}$. The system-wide instability value for the jth coordinate component S_j is defined as $S_j = \frac{1}{n}\sum_{i=1}^{n} s_{ij}$ and the system-wide instability value S is defined as $S = \sum_{i=1}^{m} S_j$.

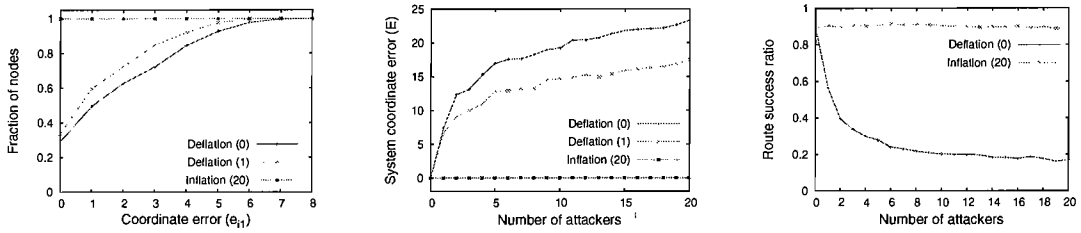
Routing performance: We characterize routing performance by using the routing success ratio which is defined as the ratio between the number of successful route requests and the number of route requests issued. Since the performance of VCS-based routing protocols relies on the success of greedy forwarding for the majority route requests, we consider only the greedy routing success in the route success ratio. We also examine the cost of the protocol by measuring the total network traffic required to route a packet.

B. Coordinate Inflation and Deflation Attacks

In these experiments, we consider that inflation and deflation attacks are performed by the attacker nodes en-route, by increasing or decreasing the hop-counts carried in the beacon packets. The beacons are assumed to behave correctly. For the coordinate deflation attack, we experiment with the case where the falsely claimed hop count is 0 and 1, referred to as Deflation(0) and Deflation(1). For the inflation attack we experiment with the case where the falsely claimed hop count is 20 referred to as Inflation(20). Deflation(0) and Inflation(20) correspond to the strongest deflation and inflation attacks. We consider two scenarios:

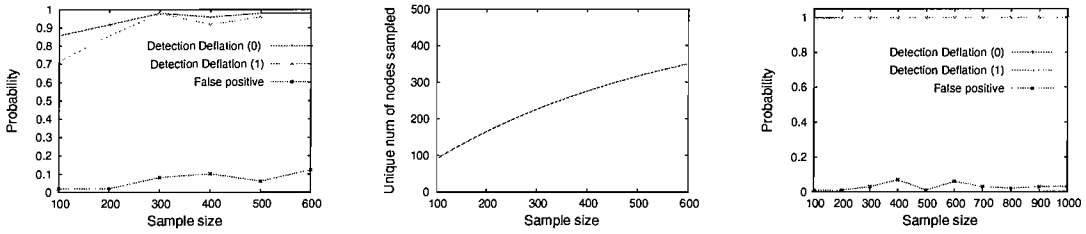
- Single beacon: This scenario investigates the impact of the attacks for one coordinate component. The coordinates modified are issued by one beacon. We vary the number of adversaries from 1 to 5, but we include only the results for the case of 5 adversaries due to their similar trend.
- Full-scale: This scenario investigates the impact of the attacks for all coordinate components. The coordinates modified are issued by all the beacons, the number of attackers varies from 1 to 20.

Fig 1(a) shows the CDF of the error of the affected coordinate component (e_{i1}) in the single beacon scenario. As it can be seen in the strongest attack, Deflation(0), about 70% of the nodes have an error greater than 1 hop and about 50% of the nodes have an error greater than 2 hops. Fig 1(b) and 1(c) show the system coordinate error E and the routing success ratio, respectively, in the full-scale scenario. The figures show that Deflation(0) has the most severe impact on both the accuracy of the virtual coordinates and the routing success ratio. A mere



(a) CDF of the coordinate error (b) System coordinate error (c) Route success ratio

Fig. 1. Impact of the coordinate inflation and deflation attacks



(a) Detection and false positive rate in TOSSIM

(b) Unique number of nodes sampled for different sample sizes in a 500 nodes network

(c) Detection and false positive rate in ideal network

Fig. 2. Wormhole-based deflation attacks detection

2 attacker nodes can bring the route success ratio from around 90% to only 40%. In contrast, the coordinate inflation attack has virtually no impact on either of these metrics. In addition, the full-scale scenario shows that the impact of the attack increases more slowly when the number of attackers goes beyond 5. This is because the network is almost fully disturbed with 5 attackers. Increasing the number of attackers further causes only limited additional damages.

The authentication techniques presented in Section IV-B.1 prevent the inflation and deflation attacks in the absence of wormholes. In the next section, we evaluate the effectiveness of our algorithm as described in Section IV-B.2 for detecting wormhole based attacks.

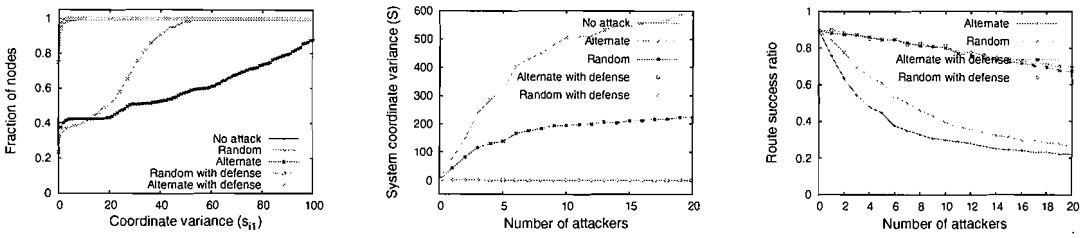
C. *Wormhole-based Coordinate Deflation Detection*

We focus on the detection of wormholes used in deflation attacks, because deflation attacks are the most severe attacks and authentication cannot prevent a wormhole-based coordinate deflation. As in the previous section, we consider the case when the falsely advertised coordinates are 0 and 1, and refer to the attacks as Deflation(0) and Deflation(1).

We demonstrate the effectiveness of our detection mechanisms in two scenarios. The first scenario uses the TOSSIM environment with a setup of 500 nodes and the average node degree of 20. The second scenario investigates the scalability of our scheme, in a simulator for ideal unit disk networks with 3000 nodes randomly distributed in a square area and the average node degree of 12. As our wormhole based deflation attack relies on statistical sampling, the increased network size allows us to show the effect of different sample sizes. In both cases we assign five randomly selected nodes as wormhole attackers which replay coordinate announcements wormhole tunneled from another attacker node located at one hop (in Deflation(0)) and two hop (in Deflation(1)) distance from a beacon node. We measure the detection rate and the false positive rate by averaging 50 different random network topologies.

Fig 2(a) shows both the detection rate and the false positive rate of the detection algorithm in the TOSSIM environment. Fig 2(b) shows the number of unique nodes queried for different sample sizes. As it can be seen from these figures, with a sample size of 300 (about 226 unique nodes in the sample), our detection algorithm achieves almost 100% detection rate. The theoretical false positive rate for the algorithm is 0.05, the selected detection cutoff P-value. However, we see a slight increase in the false positive rate as the sample size increases. This is because with larger sample sizes the detection algorithm becomes more sensitive to normal variations in the network. This anomaly can be countered by taking into account the normal network variation in the detection algorithm.

Fig 2(c) shows the result for the 3000 nodes case in the ideal simulator. The figure shows that the detection rate is almost 1 for even a sample size of 100 and the false positive rate is close to 0.



(a) CDF of coordinate variance (b) System coordinate variance (c) Route success ratio

Fig. 3. Impact of the coordinate oscillation attack with and without defense

D. Coordinate Oscillation Attack and Defense

We demonstrate the impact of the coordinate oscillation attack and the effectiveness of our proposed defense with the single beacon and the full-scale scenarios as described in Section V-B. We consider two types of oscillation attacks, the alternate oscillation attack where the attackers alternatively make the coordinate announcement of 0 and 20 and the random oscillation attack where the attackers make coordinate announcements uniformly randomly between 0 and 20. We refer to these two attacks as Alternate and Random, respectively. The interval between two announcements is uniformly randomly between 0 and 20 seconds. We record the coordinates of all the nodes every 100 seconds, after the initial 600 seconds warm-up to discount the instability due to system initialization.

Fig 3(a) shows the impact of the Alternate and Random attacks on the stability of the affected coordinate component with and without our defense mechanism in the single beacon scenario. Fig 3(b) and 3(c) show the impact of the Alternate and Random attacks on the whole coordinate system and on the routing success ratio with and without our defense mechanism. As seen in these figures, both the alternate and random oscillation attacks can cause significant instability to the node coordinates and degrade the routing success ratio significantly, and the Alternate attack has a more severe impact than the Random attack. Our defense mechanism successfully mitigates both types of attacks, with the system stability restored to the normal level and the route success ratio degrading gracefully with the number of attackers.

E. Coordinate Pollution Attack and Defense

In these experiments, all the eight beacon nodes also act as the coordinate servers. We randomly select three of the coordinate servers to be malicious. When the defense mechanism is deployed, each node stores its coordinates in three random servers based on the hash of its ID. A node invokes the majority voting mechanism to determine if a coordinate server is malicious if the route success ratio using the coordinates from the server falls below 0.5. The traffic scenario is that a node is randomly selected to be the source node, and issues route requests to other randomly selected nodes, once per second, after the initial 600 second warm-up period.

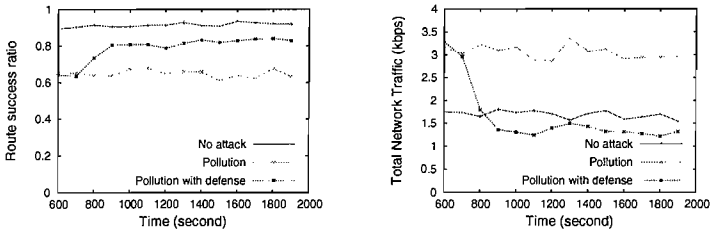
Fig 4(a) and 4(b) show the route success ratio and the total network traffic over time averaged in 100 second window for the coordinate pollution attack with and without the defense mechanism. As seen in these figures, the coordinate pollution attack not only decreases the route success ratio, but also increases the total bandwidth consumption significantly. It takes only about 100 seconds for our defense mechanism to isolate malicious beacon nodes and to return both metrics to a level similar to the no attack case. The slight discrepancy in the route success ratio between the no attack case and the attack with our defense mechanism case is due to the smaller number of honest beacons, which have a more important role in packet routing than regular nodes in BVR.

VI. Related Work

Recent work on the security of sensor networks mainly focused on proposing key management schemes that can be used to bootstrap other services [22], [23], [24], [25], [26], addressing general attacks such as Sybil [15] and replication [14] attacks, and identifying basic attacks in wireless sensor networks [11].

The problem of security in VCS-based routing protocols has not been studied to the best of our knowledge. Previous work in this area focused on improving accuracy of the virtual coordinates and the performance of routing under non-malicious environments [9] and proposing fault-tolerant techniques for VCS [6] or for the BVR routing protocol [30].

The security of geographical routing protocols using physical positions was studied in [31] for sensor networks and in [32], [33] for ad hoc networks. Most of the work focuses on preventing malicious modifications of the destination location information in packets, verifying neighbor location information, and preventing message droppings. Another main area of work in securing geographic routing is the protection of the position service in the system, which includes [34], [33]. Securing VCS-based routing protocols involves the unique challenges of securing the coordinate establishment itself, which



(a) Route success ratio over time (b) Total network traffic over time

Fig. 4. Impact of the coordinate pollution attack with and without defense

is absent in the physical position based geographic routings. Thus, a new set of measures are required.

VII. Conclusion

In this work we focused on a new class of attacks against VCS-based routing protocols for sensor networks. The attacks exploit the reliance of such protocols on the accuracy and stability of virtual coordinates. We classified these attacks as coordinate inflation, deflation, oscillation, and pollution attacks. We proposed several defense and mitigation techniques addressing each of these attacks. We demonstrated the impact of the attacks and the effectiveness of our mitigation techniques using a well-known VCS-based routing protocol, BVR, and the TOSSIM simulator. Our future work includes further analyzing the identified attacks and evaluating their impact on other applications of VCS.

References

[1] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *WSNA '02*, 2002.

[2] S. Shenker, S. Ratnasamy, B. Karp, R. Govindan, and D. Estrin, "Data-centric storage in sensornets," *SIGCOMM Comput. Commun. Rev.*, 2003.

[3] I. Vasilescu, K. Kotay, D. Rus, M. Dunbabin, and P. Corke, "Data collection, storage, and retrieval with an underwater sensor network," in *SenSys '05*, 2005.

[4] C. Perkins, E. Belding-Royer, and S. Das, *Ad hoc On-Demand Distance Vector (AODV) Routing*. IETF - Network Working Group, The Internet Society, July 2003. RFC3561.

[5] D. B. Johnson, D. A. Maltz, and J. Broch, *DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks. in Ad Hoc Networking*, ch. 5, pp. 139–172. Addison-Wesley, 2001.

[6] Q. Cao and T. Abdelzaher, "A scalable logical coordinates framework for routing in wireless sensor networks," in *RTSS '04*, 2004.

[7] A. Caruso, S. Chessa, S. De, and A. Urpi, "Gps-free coordinate assignment and routing in wireless sensor networks," in *INFOCOM '05*, 2005.

[8] R. Fonseca, S. Ratnasamy, J. Zhao, C. T. Ee, D. Culler, S. Shenker, and I. Stoica, "Beacon vector routing: Scalable point-to-point routing in wireless sensornets," in *NSDI '05*, 2005.

[9] K. Liu and N. Abu-Ghazaleh, "Aligned virtual coordinates for greedy routing in wsns," in *MASS '06*, 2006.

[10] P. Levis, N. Lee, M. Welsh, and D. Culler, "Tossim: accurate and scalable simulation of entire tinyos applications," in *SenSys '03*, 2003.

[11] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures," in *WSNA'03*, 2003.

[12] Q. Ren and Q. Liang, "Secure media access control (mac) in wireless sensor networks: intrusion detections and countermeasures," in *PIMRC 2004*, 2004.

[13] J. Douceur, "The Sybil Attack," in *IPTPS*, 2002.

[14] B. Parno, A. Perrig, and V. Gligor, "Distributed detection of node replication attacks in sensor networks," in *SP '05*, 2005.

[15] J. Newsome, E. Shi, D. Song, and A. Perrig, "The sybil attack in sensor networks: analysis & defenses," in *"IPSN '04"*, 2004.

[16] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures," in *First IEEE International Workshop on Sensor Network Protocols and Applications*, 2003.

[17] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Packet leashes: A defense against wormhole attacks in wireless ad hoc networks," in *"INFOCOM"*, 2003.

[18] L. Hu and D. Evans, "Using directional antennas to prevent wormhole attacks," in *"NDSS"*, 2004.

[19] J. Eriksson, S. V. Krishnamurthy, and M. Faloutsos, "Truelink: A practical countermeasure to the wormhole attack in wireless networks," in *"ICNP"*, 2006.

[20] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Rushing attacks and defense in wireless ad hoc network routing protocols," in *WiSe*, ACM, 2003.

[21] A. Perrig, R. Szewczyk, V. Wen, D. E. Culler, and J. D. Tygar, "SPINS: security protocols for sensor netowrks," in *Mobile Computing and Networking*, 2001.

[22] L. Eschenauer and V. Gligor, "A key management scheme for distributed sensor networks," in *CCS*, 2002.

[23] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *SP'03*, 2003.

[24] H. Chan and A. Perrig, "PIKE: Peer intermediaries for key establishment in sensor networks," in *INFOCOM*, 2005.

[25] W. Du, J. Deng, Y. S. Han, P. K. Varshney, J. Katz, and A. Khalili, "A pairwise key predistribution scheme for wireless sensor networks," *TISSEC*, vol. 8, no. 2, 2005.

[26] D. Liu, P. Ning, and R. Li, "Establishing pairwise keys in distributed sensor networks," *TISSEC*, vol. 8, no. 1, 2005.

[27] M. G. Zapata and N. Asokan, "Securing ad hoc routing protocols," in *WiSE '02*, 2002.

[28] D. S. Moore and G. P. McCabe, *Introduction to the Practice of Statistics*. New York: W.H.Freeman, 2003.

[29] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Trans. Program. Lang. Syst.*, vol. 4, no. 3, pp. 382–401, 1982.

[30] L. Demoracski, "Fault-tolerant beacon vector routing for mobile ad hoc networks," in *IPDPS '05*, 2005.

[31] N. Abu-Ghazaleh, K.-D. Kang, and K. Liu, "Towards resilient geographic routing in wsns," in *Q2SWinet '05*, 2005.

[32] T. Leinmuller, C. Maihfer, E. Schoch, and F. Kargl, "Improved security in geographic ad hoc routing through autonomous position verification," in *VANET '06*, 2006.

[33] J.-H. Song, V. W. S. Wong, and V. C. M. Leung, "Secure position-based routing protocol for mobile ad hoc networks.," *Ad Hoc Networks*, vol. 5, no. 1, pp. 76–86, 2007.

[34] X. Wu and C. Nita-Rotaru, "On the security of distributed position services," in *SecureComm 2005*, 2005.