Mitigating Attacks against Virtual Coordinate Based Routing in Wireless Sensor Networks

Jing Dong Kurt E. Ackermann Brett Bavar Cristina Nita-Rotaru

Department of Computer Science, Purdue University
305 N. University St., West Lafayette, IN 47907 USA
{dongj,keackerm,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 behave correctly. However, many applications require deploying sensor networks in adversarial environments, making VCS-based routing protocols vulnerable to numerous attacks.

In this paper, we study the security of VCS-based routing protocols. We first identify novel attacks targeting the underlying virtual coordinate system. The attacks can be mounted with little resource, yet are epidemic in nature and highly destructive to system performance. We then propose lightweight defense mechanisms against each of the identified attacks. Finally, we evaluate experimentally the impact of the attacks and the effectiveness of our defense mechanisms using a well-known VCS-based routing protocol, BVR.

Categories and Subject Descriptors

C.2.m [**Computer-communication Networks**]: Miscellaneous—*Security*; C.2.2 [**Network Protocols**]: Routing Protocols

General Terms

Security

Keywords

Sensor network routing, security, virtual coordinate system, routing, beacon vector routing, secure beacon vector routing

1. INTRODUCTION

Wireless sensor network designs have evolved from primarily focusing on data collection [28] to more sophisticated tasks such as data-centric storage [36, 39]. Likewise, the requirements for communication protocols have also evolved, from basic many-to-one and one-to-many communications to more sophisticated point-to-point communications. Well-known point-to-point wireless protocols such as AODV [32] and DSR [21] do not meet the constraints of wireless 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 VCS-based routing, each node obtains a virtual (or logical) coordinate through a virtual coordinate establishment mechanism and routing is performed in a greedy manner based on the virtual coordinates. Such routing protocols require only local interactions and minimal state information that does not grow with the size of the network. As a result, they have increased scalability and reduced overhead.

Although several VCS-based routing protocols [5,6,16,26] have been proposed in the last few years, 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 ensure these protocols operate correctly in the presence of attackers.

In addition to conventional attacks against sensor networks such as injecting, modifying, replaying, and dropping packets, VCS-based routing protocols are vulnerable to new attacks that target the virtual coordinate system. In this paper, we study the security of VCS-based routing, focusing on the unique threats that exploit the underlying virtual coordinate system. Addressing such threats is a necessary component of securing VCS-based routing and complements other techniques for addressing more general attacks on sensor networks [22]. Our contributions are:

- We identify attacks against virtual coordinate systems, and categorize them as coordinate *deflation*, *inflation*, *oscillation*, *disruption*, and *pollution* attacks. We characterize the epidemic nature of the attacks that allows even a small number of attackers to significantly degrade performance.

- We propose several novel defense mechanisms against the identified attacks. We use a statistical test based detection algorithm, hop-by-hop authentication, and a novel replay detection and mitigation technique to address coordinate deflation attacks. We stabilize the coordinate system by leveraging results from control theory to mitigate coordinate oscillation attacks, and introduce random triangle routing to mitigate coordinate disruption attacks. All of our techniques are lightweight, thus are well suited for resource constrained sensor networks.

- We evaluate through simulations the impact of the attacks and the effectiveness of our defense mechanisms, using a well-known VCS-based routing protocol, BVR [16], in the TOSSIM [24] simulator. The results validate the epidemic nature of the attacks. For example, a single coordinate deflation attacker in a network of 100 nodes can cause nearly

WiSec'08, March 31–April 2, 2008, Alexandria, Virginia, USA.

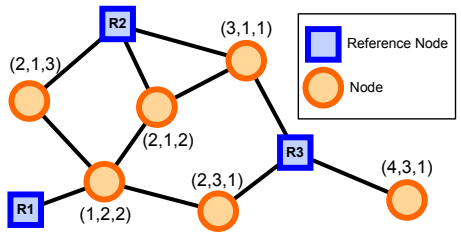


Figure 1: Virtual coordinates of nodes in a simple network

50% of nodes to have a large coordinate error (5 or larger) and a 35% reduction in the route success ratio. Our defense mechanisms successfully mitigate all the identified attacks.

Roadmap: Section 2 provides an overview of the main components of a VCS-based routing protocol. Section 3 presents attacks against the virtual coordinate system. Section 4 presents several defense mechanisms. Section 5 demonstrates the impact of the attacks and the effectiveness of our defense mechanisms through simulations. Section 6 overviews related work and Section 7 concludes our paper.

2. OVERVIEW OF VCS-BASED ROUTING PROTOCOLS

VCS-based routing protocols are similar to geographical routing protocols that forward packets to the neighboring node that is the closest to the destination. Instead of using physical coordinates, VCS-based routing protocols use virtual or logical coordinates obtained through a virtual coordinate establishment mechanism.

Most VCS-based routing protocol designs share four major components: (1) virtual coordinate establishment, (2) destination node coordinate lookup, (3) greedy routing, and (4) fall-back procedure.

The virtual coordinate establishment is achieved based on a set of reference nodes that can be special infrastructure nodes, such as landmarks [5] or regular sensor nodes [6, 16]. A common approach is for the reference nodes to periodically broadcast a *coordinate message* in the network. The coordinate message contains a hop count field that is incremented every hop to allow other nodes to derive their hop count to the corresponding reference node. The network coordinates of a node are then the set of hop counts to each of the reference nodes. Fig. 1 shows the coordinates of nodes in an example network.

In order to route a message to a destination, the source node must be able to lookup the coordinates of the destination node. A set of coordinate servers are used to maintain the coordinates of all the nodes in the network. Every node is mapped to a coordinate server for storing its coordinate by using a globally known hash function. Nodes inform their coordinate server of any coordinate changes. To lookup a coordinate, a node sends a *coordinate query message* to the coordinate server of the target node, which sends back the requested coordinate in a *coordinate reply message*. The role of coordinate servers can also be served by the reference nodes.

After obtaining the destination coordinates, the greedy geographic routing paradigm is used to route the messages. Each node forwards the message to the neighbor that is the closest to the destination using some protocol-specific distance metric.

Finally, if greedy routing reaches a node that is closer to the destination than all of its neighbors (i.e. a local minima), a fall-back procedure is invoked. For example, in the case of the BVR protocol, the fall-back procedure redirects the message to the reference node that is closest to the destination. When the message reaches that reference node, it is flooded in the network with a limited scope as determined by the destination coordinate. Typically, the fall-back procedure incurs a significantly higher overhead than the greedy routing process, so protocols strive to invoke it as rarely as possible.

3. ATTACKS AGAINST VCS-BASED ROUTING

In this section, we present several attacks against VCS-based routing protocols. These attacks exploit specific vulnerabilities in VCS-based routing protocols by targeting the control packets in the system. They are stealthy and require minimal resources from the attacker, but can cause an epidemic effect and severe damage to the performance of the routing protocol. We first present the targets of attacks by examining the components of VCS-based routing, followed by the attacker model and the details of the attacks.

3.1 VCS Attack Targets

The efficiency of a VCS-based routing protocol relies on the successful greedy routing of the majority of packets, which requires the correct operation of the virtual coordinate establishment, destination coordinate lookup, and the greedy routing process. Thus, vulnerabilities in any of these components are potential targets for attacks. We focus on attacks specific to VCS-based routing, i.e. attacks against the virtual coordinate establishment and the destination coordinate lookup. The greedy routing process is similar to the routing process in geographical-based routing protocols, so techniques proposed to secure such protocols [1, 23, 37] can also be applied to the greedy routing process of VCS-based routing.

The main service goals of virtual coordinate establishment are the *accuracy* and *stability* of the resulting virtual coordinates. Accuracy captures the closeness of the perceived coordinates of each node to their intended coordinates. An inaccurate coordinate system can cause messages to be routed in a wrong direction, leading to the invocation of the costly fall-back procedure and ultimately route failures. Stability captures the frequency and amplitude of coordinate fluctuations. An unstable coordinate system can cause route flapping, which increases routing overhead and may even cause loops in routing and route failures.

The main service goals of the coordinate lookup are the *availability* and *correctness* of the obtained destination coordinates. The destination coordinate availability is the necessary pre-condition for the initiation of routing. An incorrect destination coordinate not only misguides the packet and causes routing failures, it can also significantly increase the overhead since misguided packets usually follow lengthy paths and result in the invocation of the costly fall-back procedure.

The severe consequences of violating any of the service goals presented above make them prime targets for attacks.

3.2 Adversarial Model

We assume that the radio links are insecure. The attackers can eavesdrop, inject, modify, and replay packets.

We assume "mote" class attackers [22] where the attacker nodes are similar in capability to legitimate nodes, but in addition attackers may also collude with each other via in-band or out-of-band communication channels such as wired connections, and establish wormholes. The attacker may compromise any legitimate node and get full control of its operation. The attacker can extract the secret keys and any other data stored in compromised nodes.

Since the focus of the paper is to study attacks specific to VCS-based routing protocols, we do not consider attacks on the physical and MAC layers, such as channel jamming. These attacks can

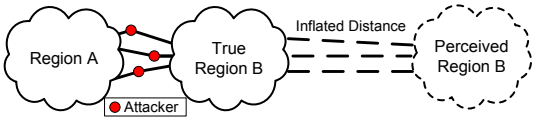


Figure 2: Example of a Virtual Coordinate Inflation Attack. The attacker nodes occupy the shortest paths from region A to region B. If the target reference node is located in region A, an artificial inflation of the coordinate by the attacker nodes can directly cause nodes in region B to derive large coordinates, and perceive themselves as far away from the target reference node.

be countered with existing techniques such as frequency hopping, spread spectrum and more resilient MAC protocols [34]. We also do not consider general attacks against sensor networks, such as Sybil [12] or node replication [31] attacks, 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 [29] and [31] are employed to address these attacks.

3.3 Attacks on Coordinate Establishment

In this section, we present attacks that aim to violate the accuracy and stability goals of the virtual coordinate establishment and categorize them based on their effect on the coordinate system. These attacks rely on manipulating the coordinate messages and can easily be performed by a low-resource "mote" class attacker.

3.3.1 Coordinate Deflation

This attack attempts to violate the coordinate accuracy property by causing legitimate nodes to obtain incorrectly small coordinates.

Since a node derives its coordinate based on its observed distance to the reference nodes, an attacker can cause incorrectly small coordinates by either directly modifying the hop count field on the coordinate messages, injecting forged coordinate messages with small hop counts, or using a wormhole which bypasses intermediate nodes to create a fictitious short path to a reference node. In message modification and injection-based attacks, the attacker nodes can either be compromised legitimate nodes or malicious outsiders that spoof legitimate nodes. In wormhole-based attacks, two or more colluding attacker nodes are required. Albeit more complex, wormhole-based attacks can circumvent authentication mechanisms that prevent the modification and injection based attacks.

3.3.2 Coordinate Inflation

This attack also targets the coordinate accuracy property as it causes legitimate nodes to obtain incorrectly large coordinates.

Similar to the deflation attack, the inflation attack can be achieved by modification or injection of coordinate messages or by wormhole tunneling a legitimate large coordinate announcement from other network regions to the local neighborhood. In addition, packet dropping by attackers on the shortest paths to a reference node can force the affected nodes to derive their coordinates via a longer path, resulting in inflated coordinates.

Unlike the coordinate deflation, the inflation attack is not always effective, as a node only uses the smallest coordinate announcement received in determining its coordinates. Thus the large coordinate announcements made by the attacker node are usually ignored. However, in network topologies where the attacker nodes occupy all the shortest paths to other network regions, as illustrated in Fig. 2, the attack is effective.

3.3.3 Coordinate Oscillation

This attack aims to violate the coordinate stability property by causing legitimate nodes to have frequent and large amplitude coordinate fluctuations.

As in the deflation and inflation attacks, the perceived distance to a reference node can be manipulated via modification, injection, dropping of coordinate messages, or wormhole tunneling. Particularly, to mount the oscillation attack, the attacker can either announce frequently changing coordinates, wormhole tunnel small and large coordinates from two or more network regions and alternately replay them in the local neighborhood, or periodically drop coordinate messages. Wormhole tunneling achieves the same effect as artificial coordinate announcement, but can circumvent authentication mechanisms that prevent artificial coordinate announcements. As in the inflation attack, the packet dropping based oscillation attack is only effective in certain network topologies where the attacker nodes occupy a vertex cut to a reference node.

3.3.4 Epidemic nature and side effects of the attacks

The above attacks are epidemic in nature. Since each node derives its coordinates based on its neighbors' coordinates, once a legitimate node is affected, it will propagate the inaccurate or unstable coordinate to its neighbors. Our simulations reveal that a single randomly placed attacker can affect as many as 50% of the nodes in a coordinate deflation attack.

Besides misguiding messages and causing route failures, the coordinate establishment attacks can also have unexpected side effects on the routing process. For example, in BVR, since messages tend to be routed toward nodes with smaller coordinates, the deflation attack also transforms attackers to powerful sinkholes that attract and control a large portion of the network traffic.

3.4 Attacks on Coordinate Lookup

In this section, we present attacks aimed at the coordinate lookup process, violating the availability and correctness of the obtained destination coordinates.

3.4.1 Coordinate Disruption

This attack prevents the querying node from receiving the coordinate reply message. Since coordinate query and reply messages also follow the greedy routing process, coordinate disruption can be caused by a malformed coordinate system or attacks on the greedy routing, such as dropping of the query or reply message by intermediate attackers.

3.4.2 Coordinate Pollution

This attack aims to cause the querying node to receive incorrect destination coordinates. It can be mounted either by compromising one or more coordinate servers which will return incorrect coordinates, by modifying the reply message during the routing process, or by forging an incorrect reply which arrives prior to the correct reply message.

4. MITIGATING VIRTUAL COORDINATE ATTACKS

In this section, we describe several mechanisms to mitigate the attacks described in Section 3. We focus on coordinate deflation, oscillation, pollution, and disruption attacks since coordinate inflation has a small impact on the network, as previously discussed and later confirmed through experiments (see Section 5). To meet the constraints of sensor networks, we adopt efficient operations such as efficient broadcast authentication, hash functions, and simple al-

gebraic manipulations, and require little state information to ensure low overhead. We adopt the principle that a node acts based only on its own observations to avoid *blacklisting* attacks and the additional overhead of trust management.

4.1 Assumptions

We consider sensor networks where nodes are primarily stationary, which is common in many applications, such as environmental, habitat, and structure monitoring. We assume that there is a period of time when the network is not under attack (e.g. the initial network deployment), which is a realistic assumption in many applications [2]. The communication between nodes is assumed to be bi-directional, though the link quality can vary significantly in the two directions.

We consider the adversarial model described in Section 3.2. In addition, like other secure routing protocols [14], we assume attackers do not form a vertex-cut in the network. Otherwise, the attackers can mount a DoS attack by simply dropping any packets passing by, which prevents any chance of successful routing across the cut.

We assume the reference nodes are trusted and also act as coordinate servers. Since they are relatively few in number, and are usually dedicated infrastructure nodes [5], special mechanisms, such as tamper-proof hardware, can be deployed to protect them. We also assume the existence of authenticated broadcast and unicast from reference nodes to other nodes in the network, which can be achieved with existing schemes, such as μTESLA [33] and predeploying shared secret keys.

4.2 Detection of Coordinate Deflation Attacks

In this section, we present a statistical test based mechanism to detect the presence of deflation attacks in the network. Our algorithm relies on the observation that the epidemic effect of the deflation attack causes a decrease of hop count for a large portion of nodes in the network. In a naive approach, one may query the coordinates of all the nodes in the network to detect such coordinate changes, and hence the presence of the attack. Instead, we propose a lightweight algorithm that does not incur any communication overhead.

Our detection algorithm uses the subset of coordinates that are already stored in the reference nodes and a popular distribution-free statistical test, the Wilcoxon signed rank test [27]. We selected the Wilcoxon test because it attains good detection rate even with a small sample set, uses paired measurements (i.e. measurements from the same samples before and after an experiment), and does not assume any underlying distribution on the measurement (e.g. normal distribution). These features suit well our application. First, using a small sample set reduces the computation overhead, and it is also desirable since reference nodes have limited storage. Second, the ability to use paired measurements allows the reference nodes to use the hop counts of the *same* set of nodes, without incurring any communication overhead for collecting extra information. Finally, the lack of assumption on the underlying distribution increases the applicability of our algorithm, as, in general, the hop count distribution is highly dependent on the network topology.

Let n be the size of the random subset of nodes whose coordinates are stored on a reference node. The algorithm performed by the reference node is as follows:

1. At a time when the network is not under attack, the reference node records its stored hop counts, $(r_1, r_2, \ldots, r_n)$, referred to as the *reference hop count*.

2. The reference node periodically compares the current hop counts stored at it $(s_1, s_2, \ldots, s_n)$ against the reference hop counts $(r_1, r_2, \ldots, r_n)$ using the Wilcoxon signed rank test, by computing

$$p = \texttt{wilc}((s_1, s_2, \ldots, s_n), (r_1, r_2, \ldots, r_n)),$$

   where `wilc` is the Wilcoxon test procedure described in the appendix. The obtained P-value p represents the probability of having the observed current hop counts given the network is not under attack.

3. We declare the network is under attack if p is less than a given threshold[1].

Since the above algorithm only uses the readily available coordinates stored on a reference node, it does not incur any extra bandwidth overhead. The main computing overhead lies in the `wilc` procedure, which involves $O(n)$ operations.

Practical issues and optimizations: In a real network, the above algorithm may raise false alarms on hop count changes due to normal network environment variations. To reduce such false alarms, we adjust the test hop count (s_i's) to account for normal hop count variations prior to applying the Wilcoxon test as follows. First, we determine a *variation compensation* (VC), which is the estimated normal hop count variation in the network. Then, prior to applying the Wilcoxon test, we set $s_i = r_i$ if s_i is smaller than r_i by no more than VC. The higher value the VC, the more robust is the algorithm in reducing false alarms. The trade-off is that a higher value for VC also has a more severe side effect of partially masking the hop count change induced by an attack, thus reducing the detection rate. The details of selecting VC are discussed in the experiment section (Section 5.4).

4.3 Preventing Deflation Attacks from Non-colluding Attackers

The unauthenticated hop count in coordinate messages presents the most severe vulnerability in the protocol, as it allows any attackers, including outsiders, to mount the deflation attack by announcing arbitrarily small hop counts. Although our above detection scheme can detect the presence of such attacks, we propose a hop count authentication scheme that eliminates such attacks from non-colluding attackers. Thus, the deflation attack may only be mounted by more powerful attackers, e.g. wormholes as discussed in Section 3.3.1.

Our hop count authentication is based on the hash chain scheme [42]. The basic hash chain technique is vulnerable to replay attacks, which can significantly reduce its effectiveness. Below, we first describe the basic hash chain mechanism in a VCS-based routing protocol, followed by the details of the replay attack and our novel mitigation technique.

4.3.1 Basic hash chain mechanism

The first step of the mechanism is the generation of the chain and distribution of the anchor used to verify the hash chain. Specifically, the reference node generates the hash chain by selecting a random number r and then applying a one-way hash function $H()$ on r iteratively N times to obtain the hash chain $v_0, v_1, v_2, \ldots, v_N$, where $v_0 = r$, $v_i = H(v_{i-1})$, and N is the estimated upper bound for the network diameter. The reference node then disseminates the tuple (v_N, N), referred to as the *anchor tuple*, throughout the network using authenticated broadcast.

[1]Typically, 0.05 is used for the threshold, which results in the commonly used confidence level of 95% for the test result.

In the coordinate establishment process, the reference node includes the tuple $(0, v_0)$, referred to as the *hop count tuple*, in its coordinate messages. When a node receives a hop count tuple (i, v_i), it first verifies that $H^{(N-i)}(v_i) = v_N$. If the verification is successful, the node determines its hop count as $i+1$ and forwards the tuple $(i+1, H(v_i))$ to its neighbors.

Since a node at hop count $i+1$ only receives the hash value v_i, assuming the hash function is pre-image resistant, it is impossible for it to generate a valid hop count tuple for any hop count less than i, thus preventing it from announcing hop count less than i to mount deflation attacks.

4.3.2 *Defending against replay attack on hash chain*

The above basic hash chain scheme is vulnerable to *replay attacks*, where an attacker node at hop count $i+1$ replays its received hop count tuple (i, v_i) to claim to be at hop count i. Our simulations (Section 5.5) demonstrate that such replay attacks can significantly degrade routing performance, primarily due to the epidemic nature of the coordinate derivation, thus it is crucial that we deploy a defense against such attacks.

For the ease of description, we refer to the neighbors of a node with smaller hop count as its *upstream nodes*, and the neighbors with larger hop count as its *downstream nodes*. We identify two variants of replay attacks, *same-distance fraud* and *transparent forwarding*. In *same-distance fraud*, the attacker reuses the received hash value to claim one smaller hop count by replaying the received packet with only the ID changed to its own or some other ID. In the *transparent forwarding attack*, the attacker transparently forwards (or tunnels) the packet unchanged to the downstream nodes, causing them to derive one smaller hop count.

Previous solutions against same-distance fraud [17] have computation and storage requirements impractical for sensor networks. None of the existing uses of hash chains [17, 19, 20, 35, 42] address the transparent forwarding problem. Instead, most of them refer to wormhole protection techniques, such as packet leashes [18], for addressing the attack. However, most wormhole protection techniques, including packet leashes, rely on special hardware, e.g. GPS, directional antenna, or tight time synchronization, which are not practical for sensor networks.

We propose a lightweight mitigation technique that addresses both variants of the replay attack without relying on any special hardware or time synchronization. Our replay defense consists of two steps, *replay detection* and *replay response*. The replay detection algorithm enables an honest node to detect the existence of a replay attacker in its neighborhood, while the replay response algorithm isolates the replay attacker from the network.

Replay detection: Our algorithm relies on the observation that a node does not know the hash value received by its upstream node. Thus, for replay detection we propose to bind the received hash value to the identity of the node, and include it in the hop count tuple.

The new hop count tuple for a node A at hop count $i, i > 0$ has the form (i, v_i, h, id), where i and v_i are for downstream nodes to verify the validity of hop count i as in the basic scheme, id is the unique ID of the node and $h = H(v_{i-1}||id)$. The new components h and id are for node A to detect any replay of its message by its neighbors. Note that a reference node may detect a replay in its neighborhood by merely overhearing any coordinate message with hop count 0.

If an attacker replays the message blindly (transparent forwarding), A will overhear his own ID broadcast from another node and detect the replay attack. If the attacker changes the id field to mask the source, he will be unable to create the hash value h as he does not know v_{i-1}. If transmitted with a garbage h value, all of its upstream nodes (who have knowledge of v_{i-1}) that overhear the packet will detect the replay by noting the hash check fails. An example is shown in Fig. 3. Note that since the reception of one replay message is sufficient for the positive detection of a replay attack, a consistent replay attacker will be detected even though some of its replay messages may not be received due to collision.

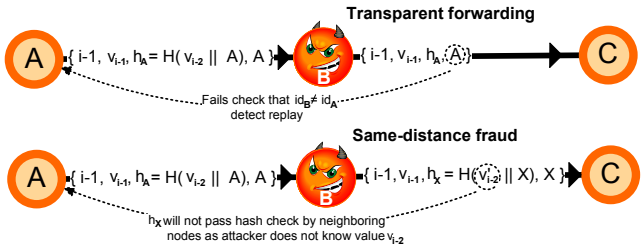


Figure 3: Replay attack detection

Replay response: We propose to isolate replay attackers by using a self sacrificing strategy. Specifically, a node that detects the existence of a replay attacker in its neighborhood voluntarily inflates its coordinate by one so that the one hop deflation of the replay attack becomes ineffective. If all the upstream neighbors of the replay attacker inflate their coordinates, the coordinate of the attacker node is forcibly inflated and thus the deflation effect of the replay attack is completely masked. Since inflating one's coordinate generally demotes one's routing priority, this scheme also has the desired effect of lowering the routing priority of the attacker node.

The drawback of this scheme is that the upstream nodes also voluntarily lower their routing priority (hence self-sacrificing). However, as demonstrated in our simulations (Section 5.5), lowering the routing priority of some nodes has little effect in the overall routing performance.

Our replay detection and response scheme requires only two additional fields (h, id) in a coordinate message, efficient hash computation, and the storage of one hash value v_{i-1} for each reference node, thus is suited well for resource constrained sensor networks.

4.4 Mitigating Coordinate Oscillation Attack

In an oscillation attack, a significant portion of honest nodes also exhibit the same behavior of oscillating coordinates as the attacker nodes due to the epidemic effect of the attack. Therefore, the naive approach of indiscriminately banning all nodes with oscillating coordinates is infeasible. Instead, a robust defense mechanism must:

- P1: Detect and isolate attackers which attack consistently
- P2: Detect and isolate strategic attackers, which can change their behavior at strategic moments or alternate between good and bad behavior
- P3: Not implicate good nodes whose behavior is affected by attacker nodes
- P4: Tolerate normal network variations

For convenience, we refer to the neighbor from which a node derives its coordinate as the node's *parent*. To defend against oscillation attacks, we leverage results from Proportional-Integral-Derivative (PID) controllers in control theory [30] to design a robust parent selection algorithm. Each node evaluates a volatility score (`VS`) for each of its neighboring nodes. Our volatility score

incorporates a node's current behavior, historical behavior, and sudden behavioral changes. A node updates the volatility score of a neighbor for every coordinate message received from the neighbor, and only uses neighbors whose VS is less than a pre-determined threshold V_T (hence not regarded as attacker) as its possible parent.

Let $\texttt{VS}_t$ denote the volatility score for a node at time t. We define

$$\texttt{VS}_t = \alpha v_t + \beta H_t + \gamma C_t, \quad (1)$$

where v_t is the current coordinate variation of the node at time t, H_t is the historical coordinate variation of the node prior to time t, and C_t is the change of coordinate variation at time t compared with previous values.

An attacker that attacks consistently will show large value for v_t and H_t, thus will be detected (P1). An attacker that only attacks at some strategic moment or alternates between good and bad behavior will show large value for v_t and C_t, thus will also be detected (P2). Once a good node picks a parent with a stable coordinate, it will also show a stable coordinate, thus will exhibit a low value for v_t, H_t, and C_t, and be regarded as good (P3). Normal network variation will only incur a small value in all three components, thus will not trigger a positive detection (P4).

The three components of the volatility score are computed as follows. We compute v_t as $v_t = c_t - c_{t-1}$, where c_i is the hop count received in the coordinate message at time i. We compute H_t as the root mean square (RMS) of all past v_t's, i.e. $H_t = \sqrt{\frac{1}{n}\sum_{i=1}^{t-1} v_i^2}$. As RMS magnifies the effect of large v_i's, it is less forgiving of bad behaviors. With some algebraic manipulation, we can obtain a recursive formulation of H_t as $H_t = \frac{1}{t}\sqrt{(t-1)H_{t-1}^2 + v_t^2}$, which allows efficient implementation that only requires the state information of the previous H_t and a count.

We compute C_t as $C_t = v_t - H_t$. Using the history H_t, instead of the previous variation v_{t-1}, produces a more stable value for C_t. To penalize sudden bad behavior and enforce slow recovery, we use different γ values, γ_1 and γ_2, for positive and negative C_t, respectively, with $\gamma_1 > \gamma_2$.

The selection of $\alpha, \beta, \gamma(\gamma_1, \gamma_2)$ determines the weights given to the node's current behavior, past behavior, and change of behavior in evaluating $\texttt{VS}_t$. The threshold value V_T determines the sensitivity of the algorithm, balancing false positive and false negative values. We describe the details of tuning these parameters in Section 5.

As no extra information is necessary, the above scheme incurs no communication overhead. The computation overhead is also minimum, involving only the evaluation of $\texttt{VS}_t$ for each received coordinate message. For storage, each node only needs store the previous coordinate announcement c_{t-1}, the historical variation H_t, and a counter for each of its neighbors.

4.5 Mitigating Attacks on Coordinate Lookup

Based on our assumption of trusted coordinate servers and authenticated unicast channel from the coordinate server to each node, we can defend against the coordinate pollution attack by authenticating coordinate replies, thus avoiding modification and spoofing of reply messages.

To mitigate coordinate disruption, we apply a *random triangle routing* technique to deliver the coordinate query and reply messages to avoid persistent routing failures. More specifically, to send a coordinate query, a node first picks a random intermediate coordinate and a range r, and delivers the query toward that coordinate. Once a node receives the query whose coordinate is within distance r of the selected intermediate coordinate, it redirects the query message to the coordinate server. The coordinate reply message follows the reverse path back to the sender.

Using a random intermediate node ensures each retrial of a failed coordinate query follows a different path, thus avoids persistent routing failures caused by consistently routing through attacker controlled regions. The drawback of this technique is the doubling of the communication overhead of the query and reply messages. However, since the destination coordinate can be cached in a node for subsequent communications, the coordinate lookup is typically invoked infrequently. For further optimization, one may use direct routing first and only resort to the random triangle routing in case of failures.

The random triangle routing technique mitigates localized attacks by routing around the attacker controlled region. Thus, it is necessary to also deploy the defense against the attacks on the coordinate establishment process, due to their potentially global impact on routing.

5. EXPERIMENTAL RESULTS

In this section, we evaluate the impact of the attacks and the effectiveness of our proposed defense based on a well-known VCS-based routing protocol, BVR [16], in the TOSSIM [24] simulator. We selected BVR because it is a mature protocol which has been shown to perform well in non-adversarial environments. In BVR, reference nodes and coordinate messages are referred to as beacons and beacon messages, respectively. Below we use this terminology to describe the experiments and discuss the results.

5.1 Experiment Setup

The network consists of 100 nodes uniformly distributed at random, with an average degree of 12, unless otherwise specified. The radio links are generated with the `LossyBuilder` tool included in TOSSIM, which generates probabilistic links based on empirical measurements from real motes. We randomly select 8 nodes to be beacons, each of which floods a beacon message at an interval uniformly from 0 to 20 seconds. For evaluating route success ratio, a routing request between two randomly selected nodes is made every second. This network setup is the default setup in the BVR source code, and has been shown to provide good performance with high fault resilience in [16]. The attackers are randomly selected and all attackers drop all data packets passing through them in order to maximize their impact on routing.

The duration of each experiment is 2000 seconds. The experiment results are the average of 10 different runs with different random topologies.

5.2 Metrics

To evaluate the impact of the deflation and inflation attacks, we characterize the accuracy of VCS with *node coordinate error* and *system coordinate error*. Let n be the number of nodes, m be the number of beacons. For node i, let h_{ib} denote the actual hop count to beacon b (as obtained when there is no attack), and x_{ib} denote the perceived hop count to beacon b. The node coordinate error for node i is defined as $e_i = \sum_{b=1}^{m} |x_{ib} - h_{ib}|$, and the system coordinate error is defined as $E = \frac{1}{n}\sum_{i=1}^{n} e_i$.

To evaluate the impact on routing, we characterize the routing performance with *routing success ratio* and *routing cost*. Routing success ratio 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 relies on the success of greedy forwarding for the majority route requests, we consider only the greedy routing success in the routing success ratio as also in [16]. To show the impact of the VCS attacks, we also compare

our attacks against the *drop only* attack where the attackers only drop data, but follow the protocol otherwise. We measure routing cost as the total network traffic required to route a packet.

5.3 Coordinate Deflation and Inflation Attacks

In these experiments, we evaluate the effect of deflation and inflation attacks on the accuracy of the virtual coordinate system and the routing performance. In the deflation attack, the attacker claims a false hop count of 0, which is the most severe form of the deflation attack. In the inflation attack, the attacker claims a false hop count of 20. Since the diameter of our experiment network is approximately 20, a false hop count of 20 represents the most severe form of the inflation attack. For each attack, we examine their effect with the number of attackers varying from 1 to 30.

Fig. 4(a) shows the CDF of the node coordinate error for the *single* attacker case. As can be seen, a single deflation attacker can cause nearly 50% of nodes to exhibit a coordinate error of 5 or larger. This network-wide impact of a single attacker validates the epidemic effect of the attack. In contrast, the inflation attack exhibits similar network coordinate variations as with the no attack case. This confirms our analysis that inflation attack is ineffective in a randomly distributed network.

Fig. 4(b) shows the average coordinate error for different number of attackers for the deflation and inflation attacks. Similar to the single attacker case, the coordinate error increases rapidly in the deflation attack as the number of attackers increases. For the inflation attack, the error increases only slightly with an increasing number of attackers. The small increase in error over the no attack case is attributed to the decrease in the density of honest nodes, rather than the impact of the attack.

Fig. 4(c) shows the impact of deflation and inflation attack on the route success ratio. The deflation attack causes a rapid decrease in the route success ratio, with 5 attackers causing the route success ratio to degrade from 90% to only 20%, as compared to 75% for the drop only attack. Such a rapid decrease in the route success ratio can be attributed to two reasons. First, the deflation attack significantly distorts the coordinate system, as evidenced by the system coordinate error (Fig. 4(b)). Second, BVR has the tendency to forward packets toward smaller coordinates, which makes deflation attackers into powerful blackholes that attract and drop traffic. We also observe that the route success ratio does not decrease much when the number of attackers increases beyond 10. This is because almost all long path routing has been disrupted by 10 or more attackers and the routing between immediate neighbors is always successful, since they are not disrupted by attackers.

On the other hand, the inflation attack has virtually no impact on the route success ratio compared to the no attack case. This is because network coordinates undergo minimal changes and large coordinate nodes (attackers) are naturally avoided in the BVR routing process.

Therefore, we conclude that deflation attacks pose severe threats to VCS routing systems, due to their epidemic effect on the coordinate system and their impact on the upper layer routing process. In contrast, inflation attacks pose little threat in a randomly distributed environment. However, we note that when the attacker nodes are located at strategic positions (e.g. vertex cut), inflation attacks can also cause severe damages.

5.4 Coordinate Deflation Detection

Since our deflation detection algorithm is based on a statistical test, to demonstrate its scalability and effectiveness, we use larger network sizes. First, we experiment with a network of 500 nodes with an average degree of 20 in TOSSIM. Then we experiment with a network of 3000 nodes with an average degree of 12 in a simulator for ideal unit disk networks.

Fig. 5 shows the detection rate of our algorithm for different variation compensation (VC) values for different sample sizes. We observe that the algorithm produces a high detection rate even when the number of attackers is small, only a small sample size is used, and the variation compensation is applied. For example, when a sample size of 50 and a variation compensation of 3 is used, the detection rate is over 95% when there are only 8 attackers (1.6% of nodes). The detection rate is more sensitive to VC when the number of attackers is smaller than 5 (1% of nodes). This is because the relatively small impact of the attack is more susceptible to statistical sampling and the masking effect of VC.

Fig 6 shows the false positive rate of the algorithm for different variation compensation and sample sizes. As can be seen, a variation compensation of 3 is sufficient to obtain a false positive rate below 3%.

In the results for the 3000 node ideal unit disk networks (Fig. 7), we observe similarly good performance of the algorithm, with the detection ratio up to 99% even when there is only one attacker for a sample size of only 50. The false positive rate is 0 for all experiments, since with no network variations the Wilcoxon test will be comparing the same set of hop counts, thus never raises false alarms.

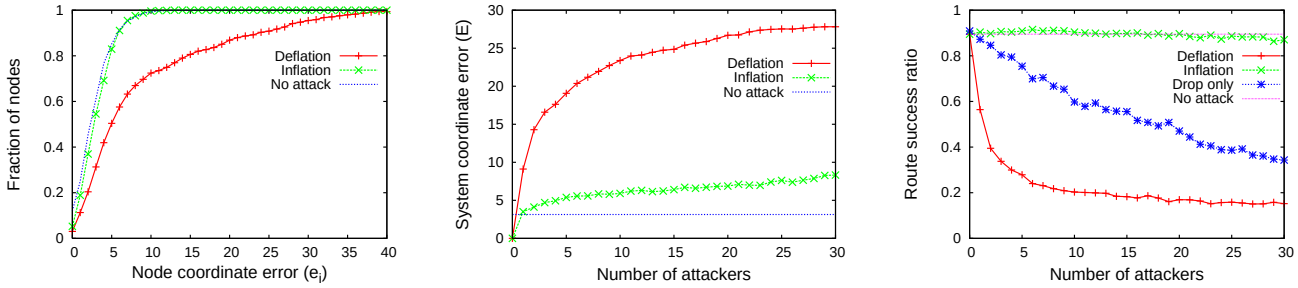
Effect of the variation compensation and sample size: We observe that the detection rate is not particularly sensitive to the increase of the variation compensation in general. This is primary due to the epidemic effect of the attack which is only masked to a limited extent by VC, and the sensitivity of the Wilcoxon test. Therefore, we can use a relatively conservative (large) VC value in order to obtain a low false positive rate.

The detection rate is also not sensitive to the sample size. A sample size of 50 is sufficient to obtain high detection ratio, thus increasing the sample size has only limited improvement on the detection ratio. Therefore, our algorithm can be run on a reference node that only keeps track of the coordinates of a small number of nodes.

5.5 Hash Chain Replay Defense

We study the performance of our defense mechanism against two types of replay attacks, Selective and Indiscriminate. In Selective replay, the attacker only replays coordinate messages with coordinate smaller than its own actual coordinate. This is the common behavior of a replay attacker that aims to deflate the coordinates of its downstream nodes. In Indiscriminate replay, the attacker replays all overheard coordinate messages regardless of their contained hop count. As an honest node will sacrifice itself by inflating its coordinate on detecting the replay of its coordinate messages, Indiscriminate replay attempts to cause the defense mechanism to backfire by causing the maximum number of honest nodes to sacrifice themselves.

Fig. 8 shows the route success ratio under the replay attack with no defense, Selective replay with defense, and Indiscriminate replay with defense. First, we observe that the route success ratio decreases rapidly when no defense is deployed for replay attack, and is about 15-20% lower than the drop only attacks shown in Fig. 4(c). This demonstrates the necessity of deploying a defense against replay attacks. With our defense mechanism deployed, there is almost no impact on the route success ratio for both types of replay attacks. Therefore, we conclude that our defense mechanism effectively mitigates replay attacks and does not suffer adverse side effects even when the attacker attempts to attack the defense mechanism itself.

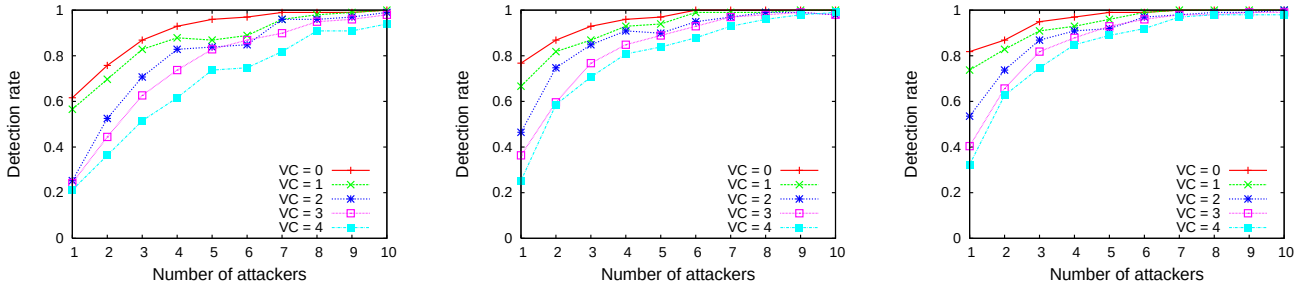


(a) CDF of the coordinate error for a single attacker

(b) Average coordinate error

(c) Route success ratio

Figure 4: Impact of the coordinate inflation and deflation attacks



(a) Sample size of 50

(b) Sample size of 150

(c) Sample size of 300

Figure 5: Detection rate of our deflation detection algorithm with different variation compensation (VC) values for different sample sizes in a 500 node network.

In terms of routing cost, we observe a similar average path length in the presence of our replay defense (Fig. 9), which demonstrates that the overhead for most packets is not affected. The slight decrease of the average path length occurs since long path packets are more likely to be dropped by attackers. However, we also observed that the flood scope of the fall-back procedure increases slightly as the number of replay attacker increases, from about 3 (no attacker case) to about 4.5 (30 attackers). This is because BVR determines the flood scope of the fall-back procedure based on the coordinates of the destination node, which are artificially inflated by our defense mechanism in the presence of attacks. Since the majority of messages are delivered in the greedy phase, the increase of the flood scope in the fall-back procedure has only limited effect on the overall routing overhead.

5.6 Coordinate Oscillation Attack and Defense

To evaluate the impact of oscillation attacks and the effectiveness of our defense mechanism, we study three different oscillation attacks: Random, Alternate, and Pulse. In the Random attack, the attacker selects a random coordinate (uniformly from 0 to 20) for each coordinate message. In the Alternate attack, the attacker alternates between two coordinate extremes (0 and 20). In the Pulse attack, the attacker oscillates the coordinate once at exponentially distributed intervals (with mean of 100 seconds) and behaves correctly at other times. The Random and Alternate attacks model consistent attackers, and the Alternate attack represents the strongest form of oscillation attack. The Pulse attack models low profile attackers that only attack at strategic moments. We use the exponential distribution to model the attack intervals, as exponential distributions are commonly used to model time intervals between events. We select 20 as the high coordinate extreme for the attack because the diameter of our experiment networks is approximately 20. The mean interval of 100 seconds in Pulse allows the effect of one attack to fade out before next attack, thus it evaluates the effectiveness of our defense in punishing sudden behavior changes.

Fig. 10 shows the route success ratio under each of the attacks, compared against the no attack and drop only attack cases. As can be seen, all three attacks result in significant degradation of the route success ratio, and are much more damaging than the drop only attack. Comparing the three attacks, Pulse poses the most dangerous threat, as this attack is of low profile (occurs only occasionally) but still causes a significant amount of damage.

Fig. 11 shows the route success ratio under each of the attacks when our defense mechanism is applied with the parameters $\alpha = 0.1, \beta = 0.9, \gamma_1 = 0.5, \gamma_2 = 0.1$, and $V_T = 6$. As can be seen, our defense mechanism restores the route success ratio to the normal level for all three attacks, which demonstrates that the defense successfully identifies and avoids both consistent and more strategic attackers, and it does not have the side effect of affecting the route success ratio due to blacklisting honest nodes.

Selection of defense parameters: We set $\alpha+\beta = 1$ and $\alpha < \beta$. Assuming an expected network variation of V_E, setting $\alpha + \beta = 1$ gives us a weighted average of the current variation v_t and the history variation H_t, obtaining approximately V_E. Setting $\alpha < \beta$ gives more weight to the history component.

As discussed in Section 4.4, we set $\gamma_1 > \gamma_2$ to penalize sudden changes in node behavior. A larger value for γ_1 penalizes sudden oscillations more severely, but an overly large γ_1 can mistakenly

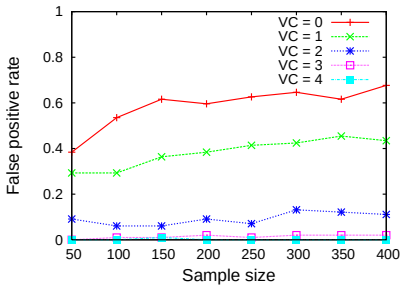


Figure 6: False positive rate of the detection algorithm for different VC and sample sizes in a 500 node network

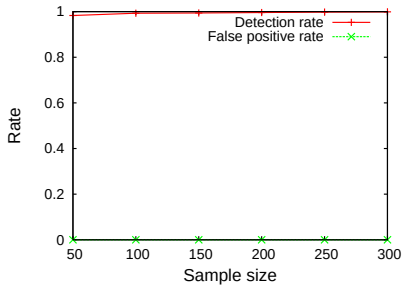


Figure 7: Detection and false positive rate of the detection algorithm in a 3000 node ideal network

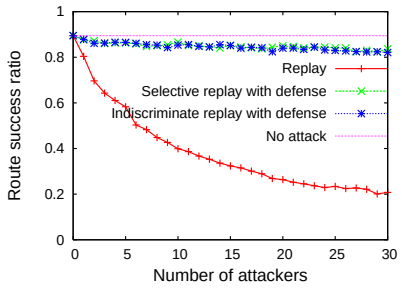


Figure 8: The effect of replay attack and our replay defense mechanism on the route success ratio

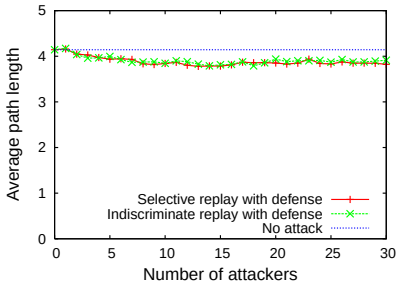


Figure 9: The effect of our replay defense mechanism on the average path length

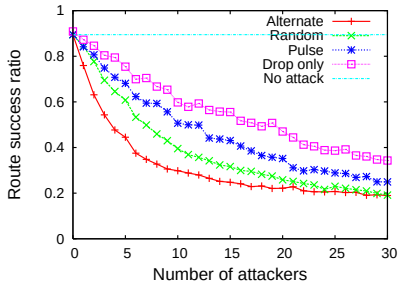


Figure 10: The impact of oscillation attacks

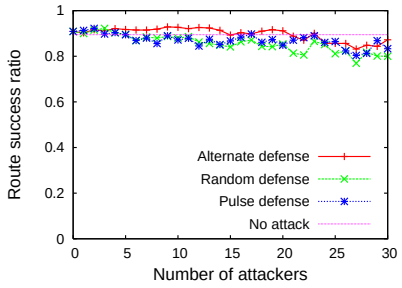


Figure 11: The effectiveness of the oscillation defense mechanism

penalize good nodes experiencing normal network variations. We empirically select $\gamma_1 = 0.5$, which is a good balance between penalizing malicious sudden behavior changes and tolerating normal network variations. γ_2 is selected small (0.1) to enforce slow recovery after an attack.

Since Equation (1) results in approximately the expected network variation V_E, we set the threshold V_T to be V_E plus a safety buffer to tolerate normal network variations. In our experiments, we observe the normal hop count variation is around 4, thus adding a buffer of 2 we obtain 6 for V_T.

6. RELATED WORK

Recent work on the security of sensor networks has focused on proposing key management schemes that can be used to bootstrap other services [7, 8, 13, 15, 25], addressing general attacks such as Sybil [29] and replication [31] attacks, and identifying basic attacks in wireless sensor networks [22].

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 [26], and proposing fault-tolerant techniques [5, 11].

The problem of securing VCS has been studied in wired networks [41]. However, the targeted VCS, Vivaldi [10], is based on an entirely different architecture that is not applicable to sensor networks, and is intended for estimating RTT, rather than routing. The solution relies on the correlation of metrics probed from random nodes, which is costly in WSNs.

The security of geographical routing protocols using physical positions was studied in [1] for sensor networks and in [23, 37] for ad hoc networks. Most of the work focuses on preventing malicious modifications of the destination location in packets, verifying neighbor location information, and preventing message dropping. Another main area of work in securing geographic routing is the protection of the position service in the system, which includes [40], [37]. Securing VCS-based routing protocols involves the unique challenge of securing the coordinate establishment itself, which is absent in physical position based geographic routing.

VCS-based routing also shares some similarity with traditional ad hoc routing protocols, such as AODV [32] and DSR [21], in that hop counts (or metrics) are accumulated hop by hop. However, in VCS-based routing, the inherent structure of the coordinate system, as well as the availability of centralized coordinate servers, allows for lightweight defense against coordinated attackers, such as wormholes, which most proposals for securing traditional ad hoc routing either do not address [17, 19, 42] or address with heavy-weight schemes such as multi-path routing and end-to-end ACK [3, 14]. In addition, since VCS-based routing is primarily designed for sensor networks, defense strategies that use specialized hardware (e.g. GPS for detecting wormholes) [18] or require intensive computation, e.g. for cryptographic signatures, or large storage [17, 19] cannot be applied. Finally, VCS-based routing has the unique component of coordinate lookup which also needs to be secured.

In our solutions, the self-sacrificing replay response shares spirit with the interesting idea of "suicide for the common good" [9], however this paper addresses the certificate revocation problem, and rely on the circulation of "suicide notes." Statistical-based tests have also been previously used in wireless network security, such as χ^2 test used in [4]. Our choice of Wilcoxon test is based on the particular characteristics and requirements of VCS. The PID-based controller theory has been previously used in reputation systems [38]. In our context, both the goals and the formulations are different.

7. 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 underlying virtual coordinate system. We classified these attacks as coordinate inflation, deflation, oscillation, disruption, and pollution attacks, and 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.

8. REFERENCES

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

[2] R. Anderson, H. Chan, and A. Perrig. Key infection: Smart trust for smart dust. In *Proceedings of IEEE International Conference on Network Protocols (ICNP 2004)*, Oct. 2004.

[3] B. Awerbuch, R. Curtmola, D. Holmer, C. Nita-Rotaru, and H. Rubens. ODSBR: An on-demand secure Byzantine resilient routing protocol for wireless ad hoc networks. To appear in *ACM Transactions on Information and System Security (TISSEC)*.

[4] L. Buttyán, L. Dóra, and I. Vajda. Statistical wormhole detection in sensor networks. In *ESAS*, 2005.

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

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

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

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

[9] J. Clulow and T. Moore. Suicide for the common good: a new strategy for credential revocation in self-organizing systems. *SIGOPS Oper. Syst. Rev.*, 40(3):18–21, 2006.

[10] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: a decentralized network coordinate system. In *SIGCOMM '04*, 2004.

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

[12] J. Douceur. The Sybil Attack. In *IPTPS*, 2002.

[13] 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*, 8(2), 2005.

[14] J. Eriksson, M. Faloutsos, and S. V. Krishnamurthy. Routing amid colluding attackers. In *ICNP '07*, 2007.

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

[16] 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.

[17] Y.-C. Hu, D. B. Johnson, and A. Perrig. Sead: Secure efficient distance vector routing in mobile wireless ad hoc networks. In *WMCSA '02*, June 2002.

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

[19] Y.-C. Hu, A. Perrig, and D. B. Johnson. Ariadne: a secure on-demand routing protocol for ad hoc networks. *Wirel. Netw.*, 11(1-2):21–38, 2005.

[20] Y.-C. Hu, A. Perrig, and M. Sirbu. Spv: secure path vector routing for securing bgp. *SIGCOMM Comput. Commun. Rev.*, 34(4), 2004.

[21] 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*, chapter 5, pages 139–172. Addison-Wesley, 2001.

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

[23] T. Leinmüller, C. Maihöfer, E. Schoch, and F. Kargl. Improved security in geographic ad hoc routing through autonomous position verification. In *VANET '06*, 2006.

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

[25] D. Liu, P. Ning, and R. Li. Establishing pairwise keys in distributed sensor networks. *TISSEC*, 8(1), 2005.

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

[27] R. Lowry. *Concepts and Applications of Inferential Statistics*, chapter 12a. Vassar College, 2006.

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

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

[30] H. Ozbay. *Introduction to Feedback Control Theory*. CRC Press, Inc., Boca Raton, FL, USA, 1999.

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

[32] 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.

[33] 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.

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

[35] S. Roy, V. G. Addada, S. Setia, and S. Jajodia. Securing maodv: Attacks & countermeasures. In *SECON 05*, September 2005.

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

[37] 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*, 5(1):76–86, 2007.

[38] M. Srivatsa, L. Xiong, and L. Liu. Trustguard: countering vulnerabilities in reputation management for decentralized overlay networks. In *WWW '05*, 2005.

[39] 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.

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

[41] D. Zage and C. Nita-Rotaru. On the accuracy of

decentralized virtual coordinate systems in adversarial networks. In *CCS '07*, 2007.

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

APPENDIX

A. WILCOXON SIGNED RANK TEST PROCEDURE

`wilc`$((s_1, s_2, \ldots, s_n), (r_1, r_2, \ldots, r_n))$

1. Compute $Z_i = r_i - s_i$. Exclude Z_i that is 0, and order non-zero absolute values $|Z_i|$ to obtain the rank R_i for each ordered $|Z_i|$. Since Z_i's are small integers, the sorting can be done in $O(n)$ using bucket sort.
2. Let N be the number of non-zero Z_i's and ϕ be the indicator function with value 1 and -1, compute $W = \sum_{i=1}^{N} \phi(Z_i) R_i$.
3. If the network is not under attack, the statistic W follows normal distribution with mean 0 and standard deviation $\sigma_W = \sqrt{\frac{N(N+1)(2N+1)}{6}}$. Thus, we can obtain the P-value p for the obtained W based on the expected normal distribution.
4. Return p.