Secure and Robust Virtual Coordinate System in Wireless Sensor Networks

JING DONG, KURT E. ACKERMANN, BRETT BAVAR
and CRISTINA NITA-ROTARU
Purdue University

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 article, we study the security of VCS-based routing protocols, with a focus on the unique component of VCS-based routing protocols, the virtual coordinate system. We first identify the security requirements of a correctly functioning VCS-based routing protocol and a set of novel attacks that can result in the violation of each of the identified requirements. The attacks target the underlying virtual coordinate system and can be mounted with low resources. However, they are epidemic in nature and are highly destructive to system performance. We then propose lightweight defense mechanisms designed specifically for resource-constrained sensor networks against each of the identified attacks. The proposed techniques require only local information on sensor nodes and take into account the unreliable nature of wireless links and network churn. 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. Our experiments show that the proposed techniques successfully mitigate all the identified attacks under a realistic link model and even at a high level of network churn.

Categories and Subject Descriptors: C.2.m [**Computer-Communication Networks**]: Miscellaneous; C.2.2 [**Computer-Communication Networks**]: Network Protocols—*Routing protocols*

General Terms: Security

Additional Key Words and Phrases: Sensor network routing, security, virtual coordinate system, routing, beacon vector routing, secure beacon vector routing

ACM Reference Format:

Dong, J., Ackermann, K. E., Bavar, B., and Nita-Rotaru, C. 2010. Secure and robust virtual coordinate system in wireless sensor networks. ACM Trans. Sensor Netw. 6, 4, Article 29 (July 2010), 34 pages. DOI = 10.1145/1777406.1777408 http://doi.acm.org/10.1145/1777406.1777408

This is a significantly enhanced version of a preliminary work that was presented at ACM WiSec'08.
Authors' addresses: J. Dong (corresponding author), K. E. Ackermann, B. Bavar, C. Nita-Rotaru, Department of Computer Science, Purdue University, West Lafayette, IN 47907; email: dongj@cs.purdue.edu.

DOI 10.1145/1777406.1777408 http://doi.acm.org/10.1145/1777406.1777408

1. INTRODUCTION

Wireless sensor network designs have evolved from primarily focusing on data collection [Mainwaring et al. 2002] to more sophisticated tasks such as data-centric storage [Shenker et al. 2003; Vasilescu et al. 2005]. 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 [Perkins et al. 2003] and DSR [Johnson et al. 2001] 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 [Cao and Abdelzaher 2004; Caruso et al. 2005; Fonseca et al. 2005; Liu and Abu-Ghazaleh 2006] 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 routing 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 article, 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 [Karlof and Wagner 2003]. Our contributions are as follows.

—We identify security requirements for VCS-based routing protocols and a set of attacks targeting each of the identified security requirements. Based on their impact on the underlying virtual coordinate system, we categorize the attacks as coordinate *deflation, inflation, oscillation, disruption*, and *pollution*. We also 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, introduce random triangle routing to mitigate coordinate disruption attacks, and use redundant coordinate servers to address coordinate pollution attacks. All of our techniques require only local state information and lightweight

computation on sensor nodes, and incorporate mechanisms to explicitly address the bursty and unreliable nature of wireless links and network churn.

—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 [Fonseca et al. 2005], in the TOSSIM [Levis et al. 2003] 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 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, with a high attack detection accuracy and low false positive rate even in the presence of unreliable wireless links and at a high level of network churn.

Roadmap. Section 2 provides an overview of the main components of a VCS-based routing protocol. Sections 3 and 4 present the security requirements and attacks on VCS-based routing protocols, respectively. Section 5 presents several defense mechanisms. Section 6 demonstrates the impact of the attacks and the effectiveness of our defense mechanisms through simulations. Section 7 overviews related work and Section 8 concludes the article.

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) a 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 [Cao and Abdelzaher 2004] or regular sensor nodes [Caruso et al. 2005; Fonseca et al. 2005]. 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. Figure 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 look up 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 look up 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.

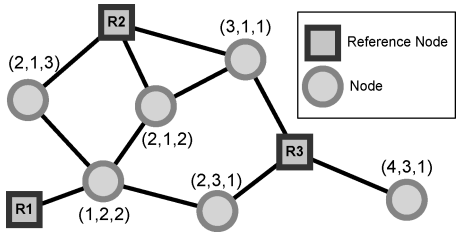


Fig. 1. Virtual coordinates of nodes in a wireless sensor network with three reference nodes.

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 [Fonseca et al. 2005], 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. SECURITY REQUIREMENTS FOR VCS-BASED ROUTING

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, the greedy routing process, and the fall-back procedure. In this section, we examine each of the components in turn and identify the critical properties of the service that must be provided by each of the components for a correctly functioning routing process. As violating any of these properties can cause severe consequences on the routing process, these properties also serve as potential targets of attacks.

Virtual coordinate establishment. 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.

Destination coordinate lookup. The main service goals of the coordinate lookup process are the *availability* and *correctness* of the obtained destination coordinates. The destination coordinate availability is the necessary precondition 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.

Greedy routing. The greedy routing process is responsible for the actual delivery of data packets using the underlying virtual coordinates. Since the process is similar to the routing process in geographical-based routing protocols, it is subject to similar types of attacks, for example, packet dropping, modification, and can similarly be protected with existing techniques proposed for securing geographical-based routing protocols [Leinmüller et al. 2006; Abu-Ghazaleh et al. 2005; Song et al. 2007].

Fall-back procedure. The fall-back procedure allows a message to escape local minima points to continue to be forwarded toward the destination. Since typically this procedure is achieved via flooding, it is more robust to attacks. In addition, it is only invoked infrequently in the normal operation of the protocol, thus attacks on this procedure can affect only a small portion of flows.

Our focus. Since the security of the greedy routing process has been addressed by existing work and attacks on the fall-back procedure have only small impact, we do not consider these two processes further in the rest of the article. Instead, we focus on attacks that are unique to VCS-based routing, that is, attacks against the virtual coordinate establishment and destination coordinate lookup process.

4. ATTACKS AGAINST VCS-BASED ROUTING

In this section, we present attacks against VCS-based routing protocols. The attacks exploit vulnerabilities specific to VCS-based routing protocols and result in violations of different properties identified in the previous section. By targeting only control packets in the system, these attacks are stealthy and require only low resources from the attacker. Yet they can cause epidemic effects and severely degrade the system performance. In the following, we first present the adversarial model, then we present the details of the attacks together with their impacts.

4.1 Adversarial Model

We assume that the radio links are inherently insecure. The attackers can eavesdrop, inject, modify, and replay packets. We assume "mote" class attackers [Karlof and Wagner 2003] where the attacker nodes are similar in capability to legitimate nodes. 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 article 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 on the MAC layer [Law et al. 2009, 2005]. These attacks can be countered with existing techniques such as frequency hopping, spread spectrum, and more resilient MAC protocols [Ren and Liang 2004]. We also do not consider attacks on upper layers, such as packet dropping attacks on

the packet forwarding process. Such attacks have been studied in previous work, such as Marti et al. [2000] and Awerbuch et al. [2008]. Finally, we do not consider general attacks against sensor networks, such as Sybil [Douceur 2002] or node replication [Parno et al. 2005] 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. Techniques such as Newsome et al. [2004] and Parno et al. [2005] can be employed to address these attacks.

4.2 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.

4.2.1 *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.

4.2.2 *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 for determining its coordinates, 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 Figure 2, the inflation of coordinates by the attackers can directly inflate the coordinates of its downstream nodes.

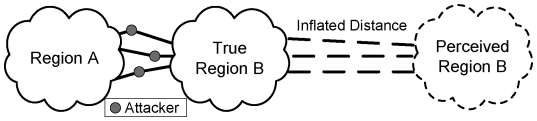


Fig. 2. Example of a virtual coordinate inflation attack. The attacker nodes occupy the shortest paths from region A to region B. If the target reference node is located in region A, an artificial inflation of the coordinate by the attacker nodes can directly cause nodes in region B to derive large coordinates,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

4.2.3 *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.

4.2.4 *Epidemic Nature and Side Effects of the Attacks.* The preceding 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.

To illustrate the severity of the epidemic effect, we analyze the percentage of nodes that can be affected by a single attacker node mounting the deflation attack against one reference node.

THEOREM 4.1. *Assuming the reference node and the attacker node are identically and independently placed (i.e., follows iid distribution) in the network area, the expected percentage of nodes affected by the attacker is 50%.*

PROOF. Let X be the random variable for the percentage of nodes affected by the attack and f_X be the pdf of X.

A node is affected by the attacker only if the node is closer to the attacker node than to the reference node. Since the reference node and the attacker node are identically and independently distributed in the network, by symmetry, we have $f_X(x) = f_X(1-x)$. Therefore,

$$\begin{aligned} E[X] &= \int_0^1 x f_X(x) dx \\ &= \frac{1}{2}\left(\int_0^1 x f_X(x) dx + \int_0^1 (1-x) f_X(1-x) dx\right) \end{aligned}$$

$$= \frac{1}{2}\left(\int_0^1 x f_X(x)dx + \int_0^1 (1-x) f_X(x)dx\right)$$
$$= \frac{1}{2}\int_0^1 f_X(x)dx$$
$$= \frac{1}{2}.$$

□

In particular, the following corollary is a special case of the previous theorem.

COROLLARY 4.2. *If the reference node and the attacker node are uniformly randomly placed in the network, the expected percentage of nodes affected by the attack is 50%.*

Our simulations also reveal that a single randomly placed attacker can affect around 50% of the nodes in a coordinate deflation attack, which corroborates the preceding analysis.

Besides misguiding messages and causing route failures, the aforesaid 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 into powerful sinkholes that attract and control a large portion of the network traffic.

4.3 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.

4.3.1 *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.

4.3.2 *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.

5. MITIGATING VIRTUAL COORDINATE ATTACKS

In this section, we describe several mechanisms to mitigate the attacks described in Section 4. 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 6). To meet the constraints of sensor networks, we ensure our defense techniques have low overhead by using efficient operations such as

efficient broadcast authentication, hash functions, and simple algebraic manipulations, and requiring little state information maintained at each node. We also adopt the principle that a node acts based only on its own observations to avoid *blacklisting* attacks, where a node spreads false rumors about other nodes, and the additional overhead of trust management.

5.1 Assumptions

We consider sensor networks where nodes are primarily stationary, which is common in many applications, such as environmental, habitat, and structural monitoring. The communication between nodes is assumed to be bidirectional, though the link quality can vary significantly in the two directions.

We consider the adversarial model described in Section 4.1. In addition, like other secure routing protocols [Eriksson et al. 2007], we assume attackers do not form a vertex-cut in the network. Otherwise, the attackers can mount a DoS (Denial-of-Service) 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 [Cao and Abdelzaher 2004], we can deploy special mechanisms, such as tamper-proof hardware, 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 [Perrig et al. 2001] and predeploying shared secret keys.

In order to achieve a secure VCS-based sensor network, it is insufficient to protect only the VCS. Instead, we emphasize that the proposed defense mechanisms for securing virtual coordinate systems are designed to be deployed in conjunction with defense schemes for protecting against attacks on other components of the system, such as Marti et al. [2000] and Awerbuch et al. [2008] for addressing packet dropping attacks.

5.2 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. However, this approach will incur a prohibitively high communication overhead for resource-constrained wireless sensor networks. 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 [Lowry 2006], to detect coordinate changes. 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.

In the following, we first present a detection scheme assuming there is a known period of time when the network is not under attack (e.g., immediately after the network is deployed). This is a realistic assumption for many applications [Anderson et al. 2004]. In the case that such a known benign period is not attainable, we present another detection scheme which instead assumes a known network topology model, for example, uniform node distribution.

5.2.1 *Attack Detection with Known Benign Period.* 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 currently stored hop counts $(s_1, s_2, \ldots, s_n)$ 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 preceding 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.

5.2.2 *Attack Detection with Known Statistical Model of Network Topology.* In this section, we propose a method for detecting deflation attacks when the statistical model of the network topology is known. The proposed method estimates the reference hop count from the known statistical model of network topology by using the Monte Carlo method [Mitzenmacher and Upfal 2005] in *offline* computation. Compared to the previous method, it eliminates the requirement of an initial known benign period without any additional overhead on the sensor nodes.

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

A statistical model of network topology includes the statistical property of the node distribution in the network and the statistical model of wireless links, which captures the availability and quality of wireless links. The statistical property of node distribution is usually available based on how the network is deployed. For example, if the sensor nodes are spread evenly in a field using an airplane, then a reasonable statistical distribution for modeling the node distribution is the uniform distribution. There has also been a significant amount of work on wireless link modeling both theoretically [Nikookar and Hashemi 1993; Rappaport 2002] and from empirical measurement results [Cerpa et al. 2005; Zamalloa and Krishnamachari 2007]. It is important not to confuse the statistical model of network topology with the network topology itself, which is deterministic in nature, referring to node positions and link qualities in a specific network. Selecting an appropriate statistical model for modeling a network topology is generally specific for each deployment environment and may require on-site survey and measurements, hence we consider it out of the scope of this article. Instead, in this section, we propose a method for detecting deflation attacks when the network model is known.

The proposed method relies on Monte Carlo methods to estimate the reference hop counts from the statistical network topology model. Monte Carlo methods are commonly used to derive the statistical property of some target variable in a complex system when the statistical distribution model of the system inputs is known. The general pattern of applying Monte Carlo methods is to first construct random instances of the system by repeatedly sampling from the random input space, then compute the value of the target variable in each random system instance, and finally aggregate the values for individual system instances to derive the final result.

In our context, the high-level steps for using the Monte-Carlo method to estimate the reference hop counts from a statistical network topology model are as follows. We first generate random network topology instances from the statistical network topology model, then we compute the reference hop counts in each random network topology, finally we aggregate the reference hop counts from each of the random network topology instances to obtain an estimate of the reference hop counts for the given statistical network topology model. Once the estimation for the reference hop counts is obtained, they are fed into the attack detection procedure as proposed in Section 5.2.1 for detecting attacks. Note that the estimation of the reference hop counts is performed *offline* in a powerful computer, rather than on sensor nodes.

The detailed procedure for estimating the reference hop counts from a known statistical network topology model is as follows.

(1) Repeat the following two steps until sample coordinates from R different random instances of network topologies are obtained, where R is a large number, for example, 1000.
   (a) Generate a random network topology instance from the known statistical network topology model. This can be achieved in two steps: First, we obtain node positions by sampling from the node distribution model. Second, we construct links between nodes using the wireless link model.

(b) Uniformly randomly sample n nodes in the random network topology instance, where n is the number of nodes whose coordinates are maintained at a reference node. Order their coordinates in increasing order and denote the resulting coordinates as $\vec{r}_j = (r_{1j}, r_{2j}, \ldots, r_{nj}), r_{ij} \leq r_{(i+1)j}$.

(2) Compute the reference hop counts $\vec{r}$ as the component-wise average of $\vec{r}_j$, that is,

$$\vec{r} = \frac{\sum_{j=1}^{N} \vec{r}_j}{N} = \left(\frac{\sum_{j=1}^{N} r_{1j}}{N}, \frac{\sum_{j=1}^{N} r_{2j}}{N}, \ldots, \frac{\sum_{j=1}^{N} r_{nj}}{N} \right).$$

Once we obtain the estimated reference hop counts, we can apply the attack detection procedure as presented for the known benign period case, except that the current hop counts $(s_1, s_2, \ldots, s_n)$ in step (2) also need to be sorted in increasing order.

Overhead analysis. We first note that in this method the operations performed by sensor nodes are exactly the same as those performed in the method proposed for the known benign period case (Section 5.2.1), except that sensor nodes no longer need to collect reference hop counts explicitly. Therefore, the computation and space overhead of this method is strictly less than the method proposed for the known benign period case. In the following, we analyse the computation and space requirements for estimating the reference hop counts in the offline computation performed by a powerful computer.

Let R denote the number of random network topology instances used in the Monte Carlo method, N denote the number of nodes in a network, and n denote the number of nodes used for the reference hop counts. For each random network topology, the time required to construct the topology is $O(N^2)$, as there are $O(N^2)$ edges in the network. Then we assign hop counts to all the nodes in the network, which can be accomplished in $O(N^2)$ using the Dijkstra's algorithm. Finally, we draw n random nodes and obtain their hop counts as the reference hop counts, each of which takes $O(1)$ time. Thus, the total time required is $O(R(2N^2 + n))$. Since $N > n$, we have the time complexity of the algorithm being $O(RN^2)$. The space requirement of the algorithm is only the storage space for the graph representing a random network topology, which takes $O(N^2)$ space.

5.2.3 *Practical Issues and Optimizations.* In a real network, wireless links are inherently unstable, which can induce a certain level of fluctuations on the network coordinates. Sensor nodes may also die over time and new nodes may be added to the network. As a result, even under benign network conditions, the network coordinates of individual nodes usually change over time. In the following, we first present a variation compensation mechanism that enhances the ability of the proposed attack detection algorithms to accommodate benign network changes. Then, we discuss the robustness of the attack detection algorithm in the presence of events such as benign link quality variations and network churn.

Enhancing tolerance to network variations with variation compensation. To reduce false alarms due to benign network variations, we propose to mask a

settable level of network variations prior to applying the Wilcoxon test. By adjusting the level of coordinate variations to be masked, one can adjust the ability of the algorithm to tolerate benign network variations, thus providing a knob for system administrators to fine-tune the algorithm to suit different levels of network variabilities.

The detailed procedure is 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 the test hop count $s_i = r_i$ if s_i is smaller than r_i by no more than VC. Thus, by applying a variation compensation of VC, one essentially masks all coordinate variations less than VC. Hence, the higher value the VC, the more robust the algorithm is 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 appropriate value for VC are discussed in the experiment section (Section 6.4).

Robustness with benign network variations. The proposed attack detection methods are based on a statistical method which by nature is able to tolerate a certain level of network variations. The variation compensation mechanism further enhances its ability of tolerating benign network variations. However, the trade-off is the potential loss of ability to detect attacks. In the following we argue that the proposed attack detection methods are able to achieve the best of both sides.

The key observation underlying our methods is the large discrepancy in the global network coordinate changes during benign network operations and when under attack. In a relatively dense network there are usually many redundant paths from a node to the reference nodes. Thus, a node usually is able to derive similar coordinates along many different paths. The high level of path redundancy in the network allows the overall network coordinates to remain stable in the presence of individual link quality changes and network churn. In contrast, as shown by both analysis (Section 4.2.4) and experimental results (Section 6.3), due to the epidemic nature of coordinate deflation attacks, even only a small number of attackers can cause significant changes in the network coordinates over a large number of nodes. The proposed attack detection method makes decisions based on observing the global network coordinate changes with efficient random sampling, and leverages the large discrepancy in the level of global coordinate changes to differentiate between attacks and benign network events. Thus, it is able to detect deflation attacks with high accuracy while maintaining a low false positive rate. Our experiments (Section 6.4) confirm our discussion and show that with appropriate variation compensations one can achieve the attack detection rate close to 1 while the false positive rate is maintained close to zero.

5.3 Preventing Deflation Attacks from Noncolluding 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 previous detection scheme can detect the presence of such attacks, we propose a hop count authentication scheme that eliminates such attacks from noncolluding attackers. We do not address attacks from colluding attackers, in particular wormholes as discussed in Section 4.2.1.

Our hop count authentication is based on the hash chain scheme [Zapata and Asokan 2002]. However, the basic hash chain technique is vulnerable to replay attacks, which can significantly reduce its effectiveness. Next, 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.

5.3.1 *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.

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 preimage 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. In practical system implementations, hash chains are usually constructed from standard cryptographic hash functions such as SHA-2 [Lilly 2002]. Thus, the hash function used satisfies not only the preimage resistance property, but also the second preimage resistance, and collision resistance property. We also note that constructing hash chains on sensor networks has unique challenges, such as the limited computation and storage capability and small packet size. We refer the reader to Bradford and Gavrylyako [2003; 2004] for more details on practical hash chain implementation issues.

5.3.2 *Defending against Replay Attack on Hash Chain.* The preceding 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.[2] Our simulations (Section 6.5) 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.

[2]Note that this replay attack differs from the *wormhole replay* attack where one attacker node tunnels its received message to another colluding attacker node to be replayed there. As noted earlier, our scheme does not address such wormhole attacks from colluding attackers.

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 [Hu et al. 2002] have computation and storage requirements impractical for sensor networks. None of the existing uses of hash chains [Hu et al. 2005, 2002, 2004; Roy et al. 2005; Zapata and Asokan 2002] addresses the transparent forwarding problem. Instead, most of them refer to wormhole protection techniques, such as packet leashes [Hu et al. 2003], or secure neighbor discovery[3] [Papadimitratos and Haas 2005], for addressing the attack. However, most wormhole protection techniques, including packet leashes, rely on special hardware, such as 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 Figure 3. Note that since the reception of one replay message is sufficient for the positive detection of a replay attack, a consistent replay

[3]Note that the secure neighbor discovery assumption is at least as strong as the no wormhole attack assumption, as the correct identification of neighbors implies the absence of wormholes.

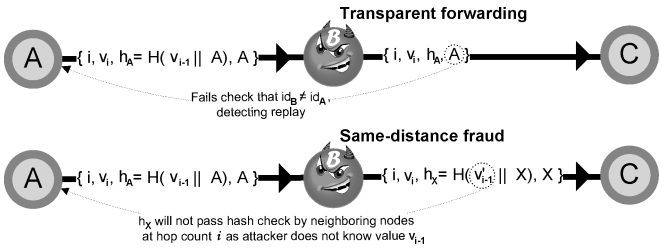


Fig. 3. Replay attack detection.

attacker will be detected even though some of its replay messages may not be received due to collision.

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, lowering the routing priority of some nodes has little effect in the overall routing performance, since there usually exists an alternative path around the node with a similar length. Our simulations in Section 6.5 also confirm the negligible effect of the self-sacrificing technique on the 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.

5.4 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 [Ozbay 1999] 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 predetermined threshold V_T (hence not regarded as attacker) as its possible parent for deriving its own coordinate. Neighbors that are detected to have a high volatility score will simply be ignored for deriving a node's coordinate.

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

$$\mathtt{VS}_t = \alpha v_t + \beta H_t + \gamma C_t, \tag{1}$$

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, that is, $H_t = \sqrt{\frac{1}{n}\sum_{i=1}^{t-1} v_i^2}$. 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 $\mathtt{VS}_t$. 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 6.

As no extra information is necessary, the preceding scheme incurs no communication overhead. The computation overhead is also minimum, involving only the evaluation of $\mathtt{VS}_t$ 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.

5.5 Mitigating Coordinate Disruption Attack

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 potential global impact on routing.

5.6 Mitigating Coordinate Pollution Attack

The authentication mechanisms described in Section 5.1 prevent modified or spoofed replies to coordinate queries, thus the main threat for coordinate pollution attacks comes from compromised coordinate servers.[4] To mitigate such threats, we have each node store its coordinates redundantly in multiple coordinate servers, instead of just one. When querying the coordinates for a target node, the node first 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 these servers for future queries.

Note that as long as the attacker does not control a majority ($\geq 50\%$) of the coordinate servers maintaining the coordinates for the target node, the previous majority voting scheme is able to return the correct coordinates. Otherwise, there is no solution [Lamport et al. 1982] guaranteeing the correctness of the returned coordinates.

Let n be the number of coordinate servers, m be the number of compromised and colluding coordinate servers, and each node stores its coordinate redundantly on k randomly selected coordinate servers. Let the probability P_f that the preceding scheme fails to return the coordinates for a node, which occurs

[4]Note that if the coordinate servers are trusted, then such threats do not exist. Here we do not assume the coordinate servers are trusted.

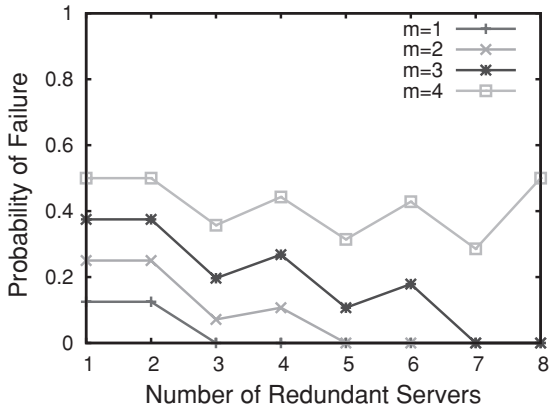


Fig. 4. The coordinate pollution defense failure probability.

when the first random coordinate server selected by the source node is malicious and the $k/2$ or more coordinate servers used to store the coordinates of the target node are malicious. Thus, we obtain P_f based on the hypergeometric distribution as

$$P_f = \sum_{i=\lceil k/2 \rceil}^{k} \frac{i}{k} \frac{\binom{m}{i}\binom{n-m}{k-i}}{\binom{n}{k}}.$$

Figure 4 plots the failure probability P_f when the number of coordinate servers is $n = 8$ for different numbers of compromised servers (m) and different numbers of redundant coordinate servers used by a node (k). First, we observe that with m malicious servers, $k = 2m + 1$ redundant servers can completely eliminate failures, which conforms to the majority voting scheme. We also observe an interesting anomaly that increasing the number of redundant servers (k) from an odd number (e.g., 3) to the next even number (e.g., 4) increases the failure probability slightly. We can explain this anomaly as follows by examining the case of $k = 2i + 1$ and $k = 2i + 2$ ($i \geq 0$), since for both $k = 2i + 1$ and $k = 2i + 2$ redundant servers, we need to have at least $i + 1$ servers being malicious in the set of redundant servers for the majority voting scheme to fail. Given the same total number of servers (n) and malicious servers (m), it is more likely to contain $i + 1$ or more malicious servers in $2i + 2$ randomly selected servers than in $2i + 1$ randomly selected servers. Hence, it is more likely for the majority voting to fail for the case of $2i + 2$ redundant servers than the case of $2i + 1$ redundant servers.

5.7 Summary on the Cost of the Defense Techniques

Bandwidth, computation, and storage are all premium resources in sensor networks, thus it is essential that any defense technique be efficient in all the aspects. Next we summarize the overhead of our defense techniques in all the three dimensions.

In the statistical-based deflation detection algorithm, there is no bandwidth overhead, as the sample coordinates are already stored in the reference nodes

for the coordinate lookup process. The reference node incurs $O(n)$ operations for the computation of the Wilcoxon test procedure and $O(n)$ storage for the reference coordinates of the sample nodes, n is the number of nodes whose coordinates are stored on the reference node. The heavy computation for estimating reference hop counts in the case of the known statistical network topology model is performed offline in a powerful computer.

The one-way hash chain-based deflation prevention incurs the same overhead as the conventional usage of one-way hash chain in wireless protocols, that is, the verification of a hash value at each node, the additional field for the hash value in coordinate messages, and the broadcast of an anchor tuple for each coordinate update. Our replay prevention technique introduces three additional fields in coordinate messages, incurs one hash verification computation cost per coordinate message received, and requires the storage of one hash value for each reference node.

In coordinate oscillation mitigation, the only overhead is the $O(1)$ computation overhead and the storage of three integer values for each of the node's neighbors. The algorithm incurs zero bandwidth overhead.

The random triangle routing for the mitigation of coordinate disruption attacks doubles the bandwidth consumption for coordinate query when under attack, but incurs no bandwidth overhead under benign conditions. There is also no computation or storage overhead.

Finally, for the coordinate pollution mitigation, the bandwidth overhead increases linearly with the number of redundant coordinate servers used. However, as coordinates are cached, the coordinate query events are infrequent, thus the total overhead remains small. The computation and storage overhead are minimum, involving only simple comparisons and storing multiple versions of coordinates for a target node.

6. 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 [Fonseca et al. 2005], in the TOSSIM [Levis et al. 2003] simulator with a realistic link model derived from empirical measurements. We selected BVR because it is a mature protocol which has been shown to perform well in nonadversarial environments. In BVR, reference nodes and coordinate messages are referred to as beacons and beacon messages, respectively. Next we use this terminology to describe the experiments and discuss the results.

6.1 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 the 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 Fonseca et al. [2005]. 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.

6.2 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 $E = \frac{1}{n} \sum_{i=1}^{n} e_i$.

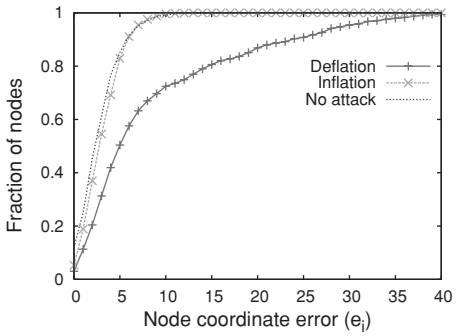
To evaluate the impact on routing, we characterize the routing performance with *routing success ratio* and *routing cost*. Routing success ratio is defined as the ratio between the number of successful route requests and the number of route requests issued. Since the performance of VCS-based routing relies on the success of greedy forwarding for the majority of route requests, we consider only the greedy routing success in the routing success ratio as also in Fonseca et al. [2005]. 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.

6.3 Coordinate Deflation and Inflation Attacks

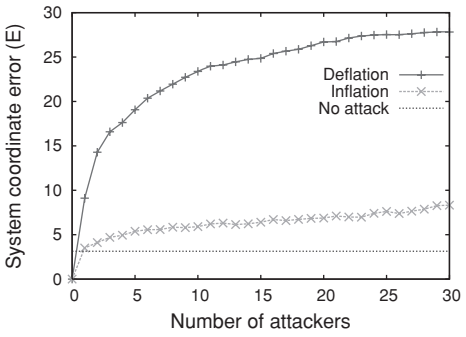
In these experiments, we evaluate the effect of deflation and inflation attacks on the accuracy of the virtual coordinate system and the routing performance. In the deflation attack, the attacker claims a false hop count of 0, which is the most severe form of the deflation attack. In the inflation attack, 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.

Figure 5(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.

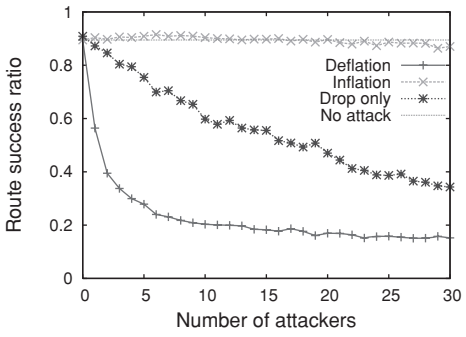
Figure 5(b) shows the average coordinate error for different number of attackers for the deflation and inflation attacks. Similar to the single attacker case, the coordinate error increases rapidly in the deflation attack as the number of attackers increases. For the inflation attack, the error increases only



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



(b) average coordinate error



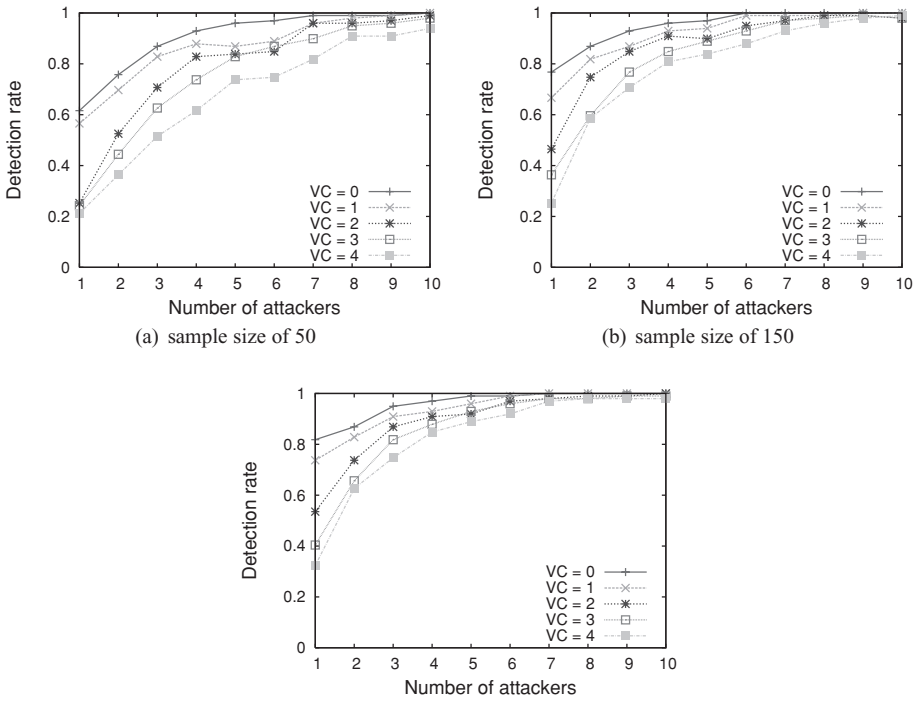
(c) route success ratio

Fig. 5. Impact of the coordinate inflation and deflation attacks.

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.

Figure 5(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 (Figure 5(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.



(a) sample size of 50 (b) sample size of 150

(c) sample size of 300

Fig. 6. Detection rate of our deflation detection algorithm with different Variation Compensation (VC) values for different sample sizes in a 500-node network.

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.

6.4 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. We experiment with a network of 500 nodes with an average degree of 20 in TOSSIM, and a network of 3000 nodes with an average degree of 12 in a simulator for ideal unit disk networks.

Attack detection with known benign period. Figure 6 shows the detection rate of our algorithm for the case of known benign period for different Variation Compensation (VC) values and 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

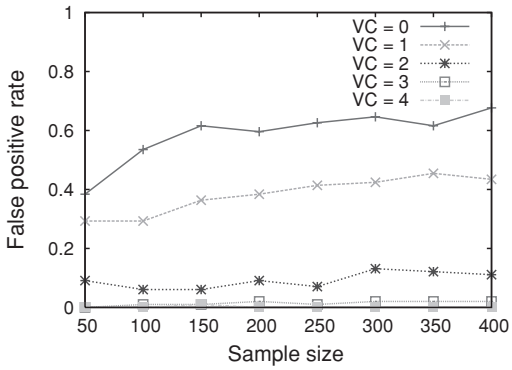


Fig. 7. False positive rate of the detection algorithm for different VC and sample sizes in a 500-node network.

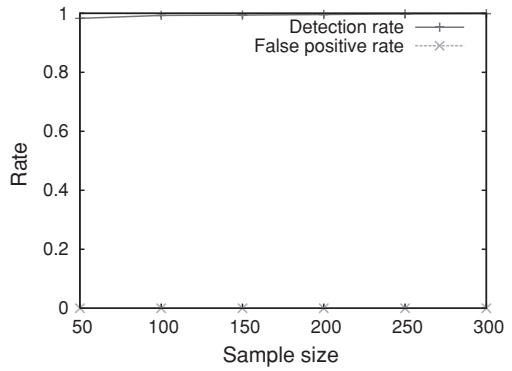


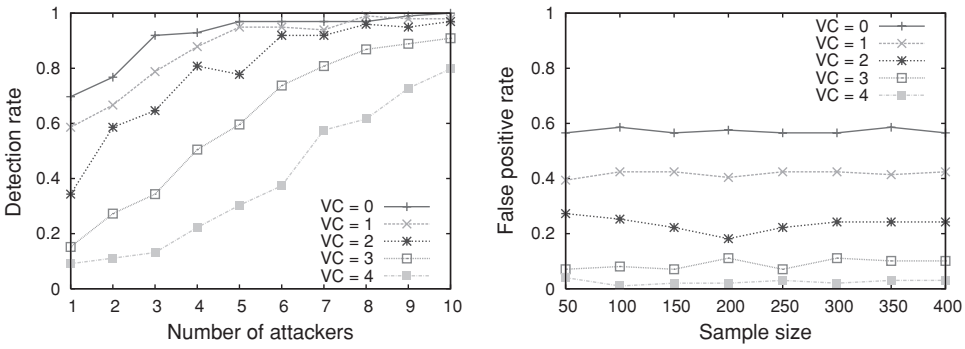
Fig. 8. Detection and false positive rate of the detection algorithm in a 3000-node ideal network.

the attack is more susceptible to statistical sampling and the masking effect of VC.

Figure 7 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 (Figure 8), 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.

Attack detection with known statistical model of network topology. Figure 9(a) and 9(b) shows the detection rate and false positive rate of our algorithm for the case of known statistical model for network topology in a 500-node network in the TOSSIM simulator. Compared with the case of known benign period, we see the effectiveness of the algorithm degrades slightly both in terms of the detection rate and the false positive rate. This is primarily due to the inaccuracy of the reference hop counts used, as the reference hop counts are



(a) Detection rate (b) False positive rate

Fig. 9. Detection rate and false positive rate of our deflation detection algorithm for the case of a known statistical model for network topology in a 500-node network in TOSSIM with the sample size of 50.

estimated rather than measured exactly, as in the previous case. However, overall the results of the two cases are similar.

Attack detection with network churn. In order to evaluate the effectiveness of the attack detection method in the presence of network churn, we repeated the TOSSIM experiments with 500 nodes under the known benign period model with various levels of churn introduced in the network. We evaluate two different types of churn that are present in realistic network deployments.

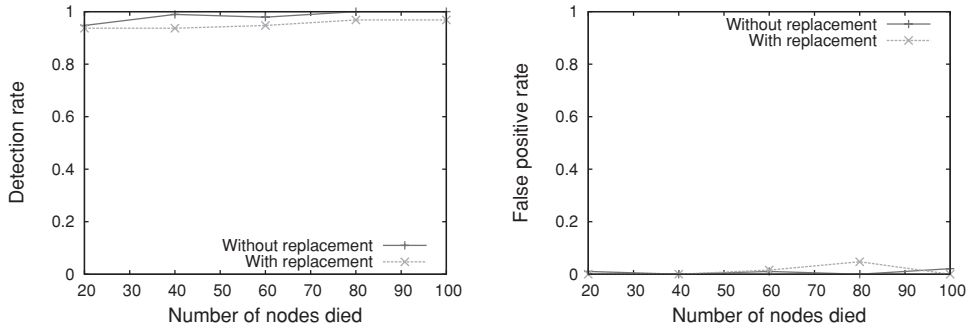
—*Removal-without-replacement*. Nodes in the network die gradually without being replaced.

—*Removal-with-replacement*. Nodes in the network die gradually and new nodes are deployed in the network in place of dead nodes.

For evaluating the case of removal-without-replacemen, we randomly remove various numbers of nodes in the network. For the case of removal-with-replacement, we randomly remove various numbers of nodes in the network and add the same number of nodes at random positions in the network, such that the network density remains the same. For both cases, we use a sample size of 150 and the variation compensation of 3, as they are also shown to work well for the experiments without considering churn. The experiment results are shown in Figure 10. As can be seen, for both removal-without-replacemen and removal-with-replacement, the proposed attack detection method is able to detect the presence of attacks with very high detection rate and maintain a close to zero false positive rate across different levels of churn in the network.

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 primarily 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



(a) detection rate with churn (b) false positive rate with churn

Fig. 10. The detection and false positive rate of the deflation attack detection method in the presence of various levels of network churn.

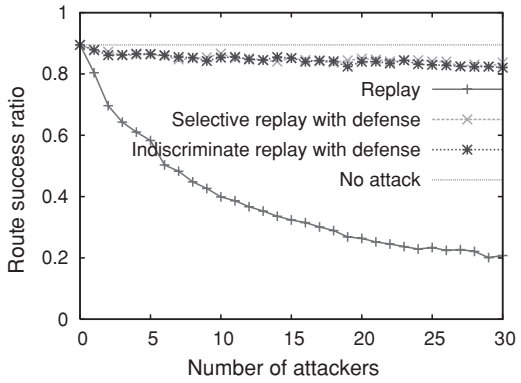


Fig. 11. The effect of replay attack and our replay defense mechanism on the route success ratio.

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.

6.5 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 a 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.

Figure 11 shows the route success ratio under the replay attack with no defense, selective replay with defense, and indiscriminate replay with defense.

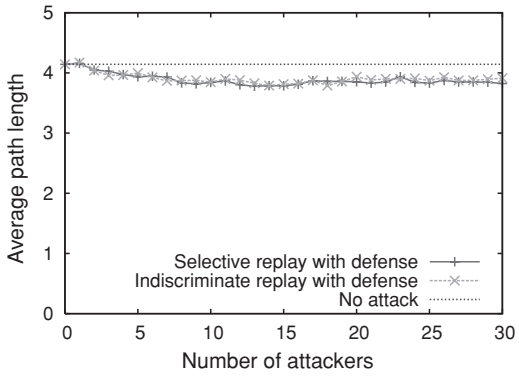


Fig. 12. The effect of our replay defense mechanism on the average path length.

First, we observe that the route success ratio decreases rapidly when no defense is deployed for replay attack, and is about 15 to 20% lower than the drop-only attacks shown in Figure 5(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.

In terms of routing cost, we observe a similar average path length in the presence of our replay defense (Figure 12), 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 attackers 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.

6.6 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

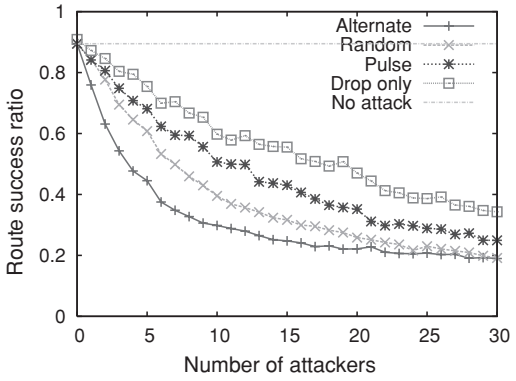


Fig. 13. The impact of oscillation attacks.

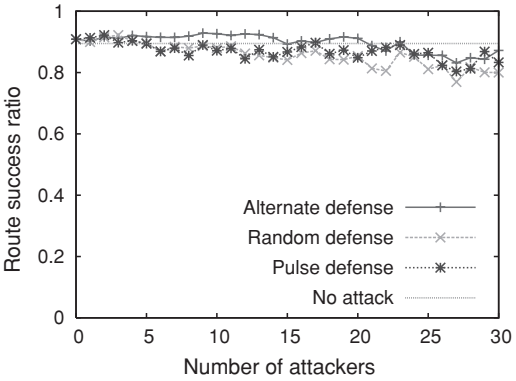


Fig. 14. The effectiveness of the oscillation defense mechanism.

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.

Figure 13 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.

Figure 14 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 $V_T = 6$. Recall that α, β, $\gamma(\gamma_1, \gamma_2)$ specify the weights given to the node's current behavior, past behavior, and change of behavior in evaluating the volatility score for a node, respectively, and V_T specifies the volatility

threshold for attack detection. 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.

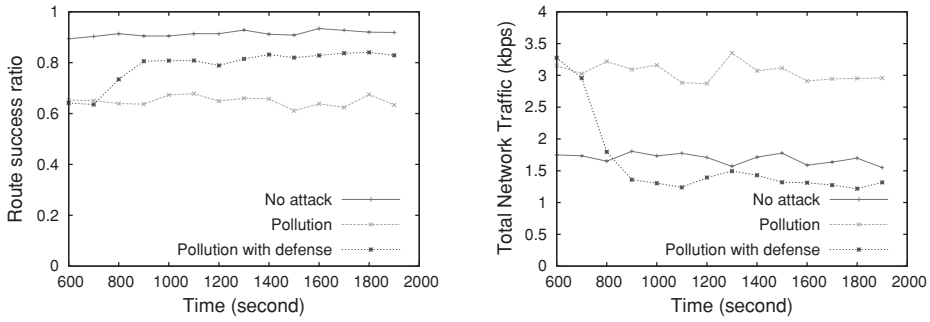
As discussed in Section 5.4, we set $\gamma_1 > \gamma_2$ to penalize sudden changes in node behavior. A larger value for γ_1 penalizes sudden oscillations more severely, but an overly large γ_1 can mistakenly penalize good nodes experiencing normal network variations. We empirically select $\gamma_1 = 0.5$, 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 Eq. (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.

6.7 Coordinate Pollution Attack and Defense

In these experiments, we evaluate the effectiveness of redundant coordinate servers in mitigating the coordinate pollution attack. The network has a total of eight coordinate servers and we randomly select three of them 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, which is sufficient for suspicion as the normal route success ratio is 0.9 as shown in the previous experiments. 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.

Figure 15(a) and 15(b) shows the route success ratio and the total network traffic over time averaged in a 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, as well as the small portion of nodes for which the defense scheme fails to return correct coordinates as a majority of its coordinate servers happen to be malicious.



(a) route success ratio over time (b) total network traffic over time

Fig. 15. Impact of the coordinate pollution attack with and without defense.

7. 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 [Eschenauer and Gligor 2002; Chan et al. 2003; Chan and Perrig 2005; Du et al. 2005; Liu et al. 2005], addressing general attacks such as Sybil [Newsome et al. 2004] and replication [Parno et al. 2005] attacks, and identifying basic attacks in wireless sensor networks [Karlof and Wagner 2003].

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 nonmalicious environments [Liu and Abu-Ghazaleh 2006], and proposing fault-tolerant techniques [Cao and Abdelzaher 2004; Demoracski 2005].

The problem of securing VCS has been studied in wired networks [Zage and Nita-Rotaru 2007a]. However, the targeted VCS, Vivaldi [Dabek et al. 2004], 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 prohibitively expensive to achieve in WSNs.

The security of geographical routing protocols using physical positions was studied in Abu-Ghazaleh et al. [2005] for sensor networks and in Leinmuller et al. [2006] and Song et al. [2007] 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 Wu and Nita-Rotaru [2005], and Song et al. [2007]. 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 [Perkins et al. 2003] and DSR [Johnson et al. 2001], 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 [Hu et al. 2002, 2005; Zapata and Asokan 2002] or address with heavyweight schemes such as multipath routing and end-to-end ACK [Eriksson et al. 2007; Awerbuch et al. 2008]. In addition, since VCS-based routing is primarily designed for sensor networks, defense strategies that use specialized hardware (e.g., GPS for detecting wormholes) [Hu et al. 2003] or require intensive computation, for example, for cryptographic signatures, or large storage [Hu et al. 2005, 2002] 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 is similar with the interesting idea of "suicide for the common good" [Clulow and Moore 2006], however, that paper addresses the certificate revocation problem, and relies on the circulation of "suicide notes." Statistical-based methods exhibit the benefit of simplicity and low overhead, and have been previously used for security in both wired networks [Walters et al. 2009; Zage and Nita-Rotaru 2007b] and wireless networks [Buttyán et al. 2005]. Walters et al. [2009] and Zage and Nita-Rotaru [2007b] use the correlations between different network metrics to identify malicious nodes, while Buttyán et al. [2005] uses the χ^2-test to detect wormholes by detecting the anomaly induced by wormhole attackers in the number of neighbors of a node and the length of shortest paths between nodes in the network. Our deflation detection method shares the similar spirit of anomaly detection with statistical methods, but the choice of Wilcoxon test is based on the unique characteristics and requirements of virtual coordinate systems in sensor networks. Finally, the PID-based controller theory has been previously used in reputation systems [Srivatsa et al. 2005] for establishing a stable reputation value that is resilient to attacker manipulations. In our context, both the goals and the formulations are different.

8. 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. Each of these attacks can be mounted with only low resources from the attacker, yet can cause significant damage to the system performance. We then proposed several defense and mitigation techniques addressing each of these attacks. Our proposed techniques do not require special hardware and are efficient in all three dimensions of bandwidth, computation, and storage, therefore, they are well-suited for resource-constrained sensor networks. They also take into account the bursty and unreliable nature of wireless links and network churn present in sensor networks. Finally, we demonstrated the impact of the attacks and the effectiveness and efficiency of our mitigation techniques using a well-known VCS-based routing protocol, BVR, in the TOSSIM simulator. The experiment results show that our defense techniques successfully mitigate all

the identified attacks with little overhead under a realistic wireless link model and even at a high level of network churn.

APPENDIX

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 nonzero 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 nonzero 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.

REFERENCES

ABU-GHAZALEH, N., KANG, K.-D., AND ANDLIU, K. 2005. Towards resilient geographic routing in wsns. In *Proceedings of the Q2SWinet'05 Conference*.

ANDERSON, R., CHAN, H., AND PERRIG, A. 2004. Key infection: Smart trust for smart dust. In *Proceedings of the IEEE International Conference on Network Protocols (ICNP'04)*.

AWERBUCH, B., CURTMOLA, R., HOLMER, D., NITA-ROTARU, C., AND RUBENS, H. 2008. Odsbr: An on-demand secure byzantine resilient routing protocol for wireless ad hoc networks. *ACM Trans. Inform. Syst. Secur. 10*, 4, 1–35.

BRADFORD, P. G. AND GAVRYLYAKO, O. V. 2003. Foundations of security for hash chains in ad hoc networks. In *Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCSW'03)*. IEEE Computer Society, 743.

BRADFORD, P. G. AND GAVRYLYAKO, O. V. 2004. Hash chains with diminishing ranges for sensors. In *Proceedings of the International Conference on Parallel Processing Workshops (ICPPW'04)*. IEEE Computer Society, 77–83.

BUTTYAN, L., DORA, L., AND VAJDA, I. 2005. Statistical wormhole detection in sensor networks. In *Proceedings of the European Workshop on Security and Privacy in Ad Hoc and Sensor Networks (ESAS'05)*.

CAO, Q. AND ABDELZAHER, T. 2004. A scalable logical coordinates framework for routing in wireless sensor networks. In *Proceedings of the IEEE Real-Time Systems Symposium (RTSS'04)*.

CARUSO, A., CHESSA, S., DE, S., AND URPI, A. 2005. Gps-Free coordinate assignment and routing in wireless sensor networks. In *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (InfoCom'05)*.

CERPA, A., WONG, J. L., KUANG, L., POTKONJAK, M., AND ESTRIN, D. 2005. Statistical model of lossy links in wireless sensor networks. In *Proceedings of the 4th International Symposium on Information. Processing in Sensor Networks (IPSN'05)*. IEEE Press, 11.

CHAN, H. AND PERRIG, A. 2005. PIKE: Peer intermediaries for key establishment in sensor networks. In *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (InfoCom'05)*.

CHAN, H., PERRIG, A., AND SONG, D. 2003. Random key predistribution schemes for sensor networks. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P'03)*.

CLULOW, J. AND MOORE, T. 2006. Suicide for the common good: A new strategy for credential revocation in self-organizing systems. *SIGOPS Oper. Syst. Rev. 40*, 3, 18–21.

DABEK, F., COX, R., KAASHOEK, F., AND MORRRIS, R. 2004. Vivaldi: A decentralized network coordinate system. In *Proceedings of the ACM SIGCOMM Data Communications Festival (SIGCOMM'04)*.

DEMORACSKI, L. 2005. Fault-Tolerant beacon vector routing for mobile ad hoc networks. In *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS'05)*.

DOUCEUR, J. 2002. The Sybil attack. In *Proceedings of the International Workshop on Peer-to-Peer Systems (IPTPS'02)*.

DU, W., DENG, J., HAN, Y. S., VARSHNEY, P. K., KATZ, J., AND KHALILI, A. 2005. A pairwise key predistribution scheme for wireless sensor networks. *ACM Trans. Inform. Syst. Secur. 8*, 2.

ERIKSSON, J., FALOUTSOS, M., AND KRISHNAMURTHY, S. V. 2007. Routing amid colluding attackers. In *Proceedings of the Annual International Conference on Network Protocols (ICNP'07)*.

ESCHENAUER, L. AND GLIGOR, V. 2002. A key management scheme for distributed sensor networks. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS'02)*.

FONSECA, R., RATNASAMY, S., ZHAO, J., EE, C. T., CULLER, D., SHENKER, S., AND STOICA, I. 2005. Beacon vector routing: Scalable point-to-point routing in wireless sensornets. In *Proceedings of the ACM Symposium on Networked Systems Design and Implementation (NSDI'05)*.

HU, Y.-C., JOHNSON, D. B., AND PERRIG, A. 2002. Sead: Secure efficient distance vector routing in mobile wireless ad hoc networks. In *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'02)*.

HU, Y.-C., PERRIG, A., AND JOHNSON, D. B. 2003. Packet leashes: A defense against wormhole attacks in wireless ad hoc networks. In *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (InfoCom'03)*.

HU, Y.-C., PERRIG, A., AND JOHNSON, D. B. 2005. Ariadne: A secure on-demand routing protocol for ad hoc networks. *Wirel. Netw. 11*, 1-2, 21–38.

HU, Y.-C., PERRIG, A., AND SIRBU, M. 2004. Spv: Secure path vector routing for securing bgp. *SIGCOMM Comput. Comm. Rev. 34*, 4.

JOHNSON, D. B., MALTZ, D. A., AND BROCH, J. 2001. DSR: The dynamic source routing protocol for multi-hop wireless ad hoc networks. In *Ad Hoc Networking*. Addison-Wesley, Chapter 5, 139–172.

KARLOF, C. AND WAGNER, D. 2003. Secure routing in wireless sensor networks: Attacks and countermeasures. In *Proceedings of the ACM International Workshop on Wireless Sensor Networks and Applications (WSNA'03)*.

LAMPORT, L., SHOSTAK, R., AND PEASE, M. 1982. The byzantine generals problem. *ACM Trans. Program. Lang. Syst. 4*, 3.

LAW, Y., HARTEL, P., DENHARTOG, J., AND HAVINGA, P. 2005. Link-Layer jamming attacks on s-mac. In *Proceeedings of the 2nd European Workshop on Wireless Sensor Networks*. 217–225.

LAW, Y. W., PALANISWAMI, M., HOESEL, L. V., DOUMEN, J., HARTEL, P., AND HAVINGA, P. 2009. Energy-Efficient link-layer jamming attacks against wireless sensor network mac protocols. *ACM Trans. Sensor Netw. 5*, 1, 1–38.

LEINMULLER, T., MAIHOFER, C., SCHOCH, E., AND KARGL, F. 2006. Improved security in geographic ad hoc routing through autonomous position verification. In *Proceedings of the International Workshop on Vehicular Ad Hoc Networks (VANET'06)*.

LEVIS, P., LEE, N., WELSH, M., AND CULLER, D. 2003. Tossim: Accurate and scalable simulation of entire tinyos applications. In *Proceedings of the ACM SIGOPS International Conference on Embedded Networked Sensor Systems (SenSys'03)*.

LILLY, G. M. 2002. Device for and method of one-way cryptographic hashing. US Patent Number 6829355.

LIU, D., NING, P., AND LI, R. 2005. Establishing pairwise keys in distributed sensor networks. *ACM Trans. Inform. Syst. Secur. 8*, 1.

LIU, K. AND ABU-GHAZALEH, N. 2006. Aligned virtual coordinates for greedy routing in wsns. In *Proceedings of the IEEE Conference on Mobile, Ad Hoc and Sensor Systems (MASS'06)*.

LOWRY, R. 2006. *Concepts and Applications of Inferential Statistics*. Vassar College, Chapter 12a.

MAINWARING, A., CULLER, D., POLASTRE, J., SZEWCZYK, R., AND ANDERSON, J. 2002. Wireless sensor networks for habitat monitoring. In *Proceedings of the ACM International Workshop on Wireless Sensor Networks and Applications (WSNA'02)*.

MARTI, S., GIULI, T., LAI, K., AND BAKER, M. 2000. Mitigating routing misbehavior in mobile ad hoc networks. In *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking (MobiCom'00)*.

MITZENMACHER, M. AND UPFAL, E. 2005. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press.

NEWSOME, J., SHI, E., SONG, D., AND PERRIG, A. 2004. The sybil attack in sensor networks: Analysis and defenses. In *Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN'04)*.

NIKOOKAR, H. AND HASHEMI, H. 1993. Statistical modeling of signal amplitude fading of indoor radio propagation channels. In *Proceedings of the 2nd International Conference on Universal Personal Communication, Personal Communications: Gateway to the 21st Century*. 84–88.

OZBAY, H. 1999. *Introduction to Feedback Control Theory*. CRC Press, Boca Raton, FL.

PAPADIMITRATOS, P. AND HAAS, Z. 2005. Secure on-demand distance vector routing in ad hoc networks. In *Proceedings of the IEEE/Sarnoff Symposium on Advances in Wired and Wireless Communication*. 168–171.

PARNO, B., PERRIG, A., AND GLIGOR, V. 2005. Distributed detection of node replication attacks in sensor networks. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P'05)*.

PERKINS, C., BELDING-ROYER, E., AND DAS, S. 2003. *Ad hoc On-Demand Distance Vector (AODV) Routing*. IETF—Network Working Group, The Internet Society. RFC3561.

PERRIG, A., SZEWCZYK, R., WEN, V., CULLER, D. E., AND TYGAR, J. D. 2001. SPINS: Security protocols for sensor netowrks. In *Proceedings of the Conference on Mobile Computing and Networking*.

RAPPAPORT, T. 2002. *Wireless Communications: Principles and Practice*. Prentice Hall.

REN, Q. AND LIANG, Q. 2004. Secure media access control (mac) in wireless sensor networks: Intrusion detections and countermeasures. In *Proceedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'04)*.

ROY, S., ADDADA, V. G., SETIA, S., AND JAJODIA, S. 2005. Securing maodv: Attacks and countermeasures. In *Proceedings of the IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON'05)*.

SHENKER, S., RATNASAMY, S., KARP, B., GOVINDAN, R., AND ESTRIN, D. 2003. Data-Centric storage in sensornets. *SIGCOMM Comput. Comm. Rev. 33*, 1, 137–142.

SONG, J.-H., WONG, V. W. S., AND LEUNG, V. C. M. 2007. Secure position-based routing protocol for mobile ad hoc networks. *Ad Hoc Netw. 5*, 1, 76–86.

SRIVATSA, M., XIONG, L., AND LIU, L. 2005. Trustguard: Countering vulnerabilities in reputation management for decentralized overlay networks. In *Proceedings of the International World Wide Web Conference (WWW'05)*.

VASILESCU, I., KOTAY, K., RUS, D., DUNBABIN, M., AND CORKE, P. 2005. Data collection, storage, and retrieval with an underwater sensor network. In *Proceedings of the ACM SIGOPS International Conference on Embedded Networked Sensor Systems (SenSys'05)*.

WALTERS, A., ZAGE, D., AND ROTARU, C. N. 2009. A framework for mitigating attacks against measurement-based adaptation mechanisms in unstructured multicast overlay networks. *IEEE/ACM Trans. Netw. 16*, 6, 1434–1446.

WU, X. AND NITA-ROTARU, C. 2005. On the security of distributed position services. In *Proceedings of the SecureComm'05 Conference*.

ZAGE, D. AND NITA-ROTARU, C. 2007a. On the accuracy of decentralized virtual coordinate systems in adversarial networks. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS'07)*.

ZAGE, D. J. AND NITA-ROTARU, C. 2007b. On the accuracy of decentralized virtual coordinate systems in adversarial networks. In *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS'07)*. ACM, New York, 214–224.

ZAMALLOA, M. AND KRISHNAMACHARI, B. 2007. An analysis of unreliability and asymmetry in low-power wireless links. *ACM Trans. Sensor Netw. 3*, 2, 7.

ZAPATA, M. G. AND ASOKAN, N. 2002. Securing ad hoc routing protocols. In *Proceedings of the International Conference on Web Information Systems Engineering (WiSE'02)*.

Received August 2008; revised September 2009; accepted September 2009