NCShield: Securing Decentralized, Matrix Factorization-Based Network Coordinate Systems

Shining Wu[1], Yang Chen[2,†] , Xiaoming Fu[1], Jun Li[3]
[1]*Institute of Computer Science, University of Goettingen, Germany*
[2]*Department of Computer Science, Duke University, USA*
[3]*Department of Computer and Information Science, University of Oregon, USA*
E-mail:{shining.wu, fu}@cs.uni-goettingen.de, ychen@cs.duke.edu, lijun@cs.uoregon.edu

***Abstract*—While network coordinate (NC) systems provide scalable Internet distance estimation service and are useful for various Internet applications, decentralized, matrix factorization-based NC (MFNC) systems have received particular attention recently. They can serve large-scale distributed applications (as opposed to centralized NC systems) and do not need to assume triangle inequality (as opposed to Euclidean-based NC systems). However, because of their decentralized nature, MFNC systems are vulnerable to various malicious attacks.**

In this paper, we provide the first study on attacks toward MFNC systems, and propose a decentralized trust and reputation approach, called *NCShield*, to counter such attacks. Different from previous approaches, our approach is able to distinguish between legitimate distance variations and malicious distance alterations. Using four representative data sets from the Internet, we show that NCShield can defend against attacks with high accuracy. For example, when selecting node pairs with a shorter distance than a predefined threshold in an online game scenario, even if 30% nodes are malicious, NCShield can reduce the false positive rate from 45.5% to 3.7%.

I. Introduction

Network Coordinate (NC) systems have been developed to estimate distance between nodes on the Internet and assist latency-conscious Internet applications, such as application layer multicast [36], BitTorrent file sharing [30], network modeling [35], compact routing [1], multi-player online games [2], [23], network monitoring [28], social networking [11], [37] and cloud service [14], [33]. In a network of N nodes, instead of incurring $O(N^2)$ measurements to obtain all pairwise round-trip times (RTTs), NC systems only need to perform $O(N)$ RTT measurements to estimate node distance, significantly reducing measurement overheads.

NC systems follow two basic models. Most traditional NC systems (such as GNP [25], PIC [12], NPS [26], and Vivaldi [13]) are Euclidean-based NC (ENC) systems. All nodes are embedded in an Euclidean space R^d, every node is assigned a d-dimensional coordinate ($d \ll N$), and node distance is estimated by typical Euclidean distance calculation. However, ENC systems assume that estimated distances among every three hosts must satisfy the triangle inequality, a condition that often does not hold true on today's Internet [18], [20], [22], [24], [38]. The other model, which has recently received much attention, is the matrix factorization based NC (MFNC) systems [24], including IDES [24], Phoenix [9], and DMF [21]. It completely removes the triangle inequality constraint, and has shown to be more accurate than ENC systems.

An NC system can be either centralized or decentralized. As centralized NC systems, such as GNP [25] and IDES [24], rely on a small set of landmark nodes, which could easily become the scalability bottleneck, decentralized NC systems are therefore the only feasible option for serving large-scale distributed applications. We therefore focus on decentralized NC systems (such as Phoenix [9] and DMF [21]) in this paper.

Decentralization, however, makes a decentralized NC system vulnerable to certain security attacks. While every node in the system can advertise to other nodes arbitrary information at its own discretion, malicious nodes in the system can falsify coordinates or delay the response to RTT probing packets in order to disrupt an NC system.

While several approaches [17], [27], [29], [34] have been proposed to secure ENC systems, unfortunately, little has been done toward MFNC systems. We tackle this deficiency in this paper. In particular, we make the following contributions:

1) We formalize potential malicious attacks toward decentralized MFNC systems. So far, protecting MFNC systems from malicious attacks has not been considered, and no attack model for such systems exist. Also, through extensive evaluation using four representative data sets collected from the Internet, we show how these attacks can disrupt existing MFNC systems such as Phoenix [9] and DMF [21].
2) We propose a trust and reputation-based approach, called **NCShield**, to defend decentralized MFNC systems. NCShield is fully decentralized and can be easily integrated into existing MFNC systems. Instead of relying on additional infrastructures, such

† This work was done when Yang Chen was with Institute of Computer Science, University of Goettingen, Germany.

978-1-4673-1298-1/12/$31.00

as distributed hash tables (DHTs) (as in [5]) or a centralized reputation computation agent (RCA) (as in [27]), NCShield uses secure gossip [4] to ensure lightweight and unbiased node sampling when choosing nodes to calculate coordinates. Based on their scalable measurement, nodes can vote in a distributed way to identifying malicious nodes.

3) Different from previous approaches, our solution is able to distinguish between ordinary distance variation and malicious distance alteration. As demonstrated by our experiments using classic aggregate data sets and a dynamic data set, as well as an online game scenario, NCShield achieves a high estimation accuracy. For example, in the online game scenario when selecting node pairs with a shorter distance than a predefined threshold, even if 30% nodes in the system are malicious, NCShield can reduce the false positive rate from 45.5% to 3.7%.

The rest of the paper is organized as follows. We first describe the background of our work in Section II, including how decentralized MFNC systems function and how attacks and defenses of decentralized ENCS operate. In Section III, we investigate potential attacks toward MFNC systems, and present our defense approach, NCShield. Then, in Section IV we conduct extensive performance evaluation of our defense mechanism with aggregate data sets used in Phoenix and DMF systems. In Section V, we evaluate NCShield using a dynamic data set and emulate it in an online game scenario with Phoenix, showing its effectiveness and feasibility. Section VI concludes this paper.

II. Background and Related Work

In this section, we first describe decentralized MFNC systems, especially its features related to securing MFNC systems. We then present how decentralized ENC systems can be attacked by malicious nodes and how they can be protected via several existing defense approaches, in order to shed light on the security solution for MFNC systems.

A. Decentralized MFNC Systems

An NC system is expected to run in Internet scale with up to millions of Internet nodes [30]. Once a node obtains a list of neighbors, referred as *NList*, it periodically learns the coordinates of its neighbors, measures its distance to these neighbors, and updates its own coordinates. An NC system is accurate when the distance between two nodes predicted using their coordinates is very close to the measured distance.

In an MFNC system with N nodes, every node has two d-dimensional ($d \ll N$) coordinates, the outgoing vector and the incoming vector. The predicted distance from node i to node j is determined by the dot product of node i's outgoing vector and node j's incoming vector, as in Eq. 1 where $D(i,j)^E$ is the predicted distance from i to j, $\vec{X_i}$ and $\vec{Y_j}$ are i's outgoing vector and j's incoming vector, respectively:

$$D(i,j)^E = \vec{X_i} \cdot \vec{Y_j} = \sum_{k=1}^{d} x_{ik} \cdot y_{jk} \tag{1}$$

This prediction mechanism is more flexible than the Euclidean distance model as it does not need to satisfy the triangle inequality principle.

The accuracy of an NC system can be evaluated using the Relative Error (RE) calculated from the predicted distance and the measured distance [8]–[10], [13], [20], [24]–[26]. The RE of a pair of nodes i and j is defined as:

$$RE = \frac{|D^E(i,j) - D(i,j)|}{\min(D^E(i,j), D(i,j))} \tag{2}$$

A system with smaller RE has higher prediction accuracy. As in Eq. 2, RE is a non-negative indicator. If predicted distance equals to measured distance, the RE will be zero.

An MFNC system may be further inspected using an overall minimization objective function (Δ), where $D(i,j)$ denotes the measured distance from node i to node j:

$$\Delta = \|D - D^E\|^2 = \sum_i \sum_j (D(i,j) - D^E(i,j))^2 \tag{3}$$

Every node in an MFNC system will also need to update its coordinates. In DMF and Phoenix, every node, say H, adopts the least squares functions to update its new coordinates. Assuming node H has m reference nodes (i.e. neighbors), the new coordinates will minimize both the distance from H to reference nodes and the distance from reference nodes to H, as shown in the two equations below:

$$\vec{x_{new}} = arg \min_{x \in F^d} \sum_{i=1}^{m} w_{Y_i} \|D_i^{out} - \vec{x} \cdot \vec{Y_i}\|^2 \tag{4}$$

$$\vec{y_{new}} = arg \min_{y \in F^d} \sum_{i=1}^{m} w_{X_i} \|D_i^{in} - \vec{y} \cdot \vec{X_i}\|^2 \tag{5}$$

$\vec{X_i} (1 \le i \le m)$ and $\vec{Y_i} (1 \le i \le m)$ are the outgoing vectors and incoming vectors of H's reference node i. D_i^{out} is the measured distance from node H to its reference node i, and D_i^{in} is the measured distance from reference node i to node H. In DMF, the weights of neighbors are always 1 and the field F is R. In Phoenix, the weights are calculated by a weight-based algorithm and the field is R^+, indicating non-negative d-dimensional coordinates. The weight-based algorithm can improve the overall prediction accuracy by alleviating error propagation.

B. Classifications of Attacks on Decentralized ENC Systems

Several types of attacks have been shown to bring significant disruptions to ENC systems [16]. Based on their objectives, attacks on decentralized ENC systems can be classified into three categories:

Disorder attack: Malicious nodes try to disorder the entire NC system to result in low prediction accuracy or cause the system hard to converge.
Repulsion attack: Victim nodes are convinced to be far away from other participants of the system. Other nodes may thus not be willing to connect to victims.
Isolation attack: Victim nodes are convinced to be in a particular area of the network and they are isolated from ordinary nodes. They thus may more easily connect with malicious nodes.

C. Existing Defense Approaches for Decentralized ENC Systems

Researchers have proposed several approaches to defending decentralized ENC systems. Though implemented on different infrastructures, these approaches are based on a common idea: in updating node coordinates they all use extra information to determine whether or not a node is trustworthy. These approaches can be broadly classified into two categories: node behavior based approaches [17], [34] and trust and reputation based approaches [27], [29].

Kaafar *et al.* [17] propose a node behavior based approach for ENC system defense. Their basic idea is that the dynamics of a node in a normal system can be modeled by a linear state space and thus tracked by a Kalman filter. It chooses a set of dedicated nodes as trusted surveyors to observe the dynamics of nodes and maintain the parameters of Kalman filter. Malicious behaviors of a node can be identified by its neighboring surveyors using the Kalman filter. The weakness of this approach is the dedicated surveyor nodes, which produce a significant overhead when the scale of the system becomes large (e.g., 800-1000 surveyors are suggested for an NCS serving 10,000 nodes). To overcome this weakness, Zage *et al.* [34] propose a fully distributed approach without relying on a set of dedicated surveyors. It detects malicious nodes by observing inconsistent behaviors with their neighbors (temporal outlier) or the space of metrics (spacial outlier). This approach avoids extra communication overheads, but as the detection of temporal outliers depends on nodes' history information [29], it does not perform well with frequent node churns. In addition, node behavior based approaches can suffer from the complexity in measuring and analyzing node behaviors,

RVivaldi [27] and Veracity [29] are both a trust and reputation based defense approach for ENC systems. RVivaldi employs two types of entities: a centralized Reputation Computation Agent (RCA) and surveyors, where surveyors monitor nodes and RCA performs mathematical computation of a node's trust and reputation score. Clearly, the centralized RCA becomes a single point of failure since it is responsible for computing the reputation score of every node in the system. Veracity [29] does not need a centralized RCA. Instead, it employs two sets of nodes, VSet (voting node set) and RSet (reference node set), to help verify the process of updating node coordinates. It applies Distributed Hash Table (DHT) for VSet and RSet construction as well as neighbor selection. However, as shown in Section III-D, this approach requires a significant amount of communication overhead in order to maintain its overlay routing structure. In addition, it requires additional security methods such as [5] for protecting this additional infrastructure, which further adds extra overhead.

Due to the differences between MFNC systems and ENC systems, a defense for MFNC systems needs to take account of the specific feature of MFNC systems, All aforementioned approached are designed specifically for ENC systems, where attackers only target the d-dimensional coordinates of a node. In a MFNC system, however, malicious nodes have more target options than those in ENC systems; for example, they may compromise either the outgoing vector and the incoming vector of a node.

In addition, all of the four defense approaches above are only evaluated using *aggregate* data sets, in which the RTT between any two hosts in the data set is a *single* value, based on either the median [13] or the minimum of measured RTTs [35], [38] over a period of time (days or even weeks). However, the distance between nodes can vary from time to time [22]. These approaches do not address and demonstrate how common distance variation—which frequently happens on the Internet—may not be detected as a malicious distance alteration.

III. Attacks and a Defense Approach in Decentralized MFNC Systems

As described above, attacks in MFNC systems are different and more complicated. Furthermore, the use of monitoring nodes are essential to protect a decentralized system, and thus their discovery and selection algorithm should be carefully designed. With these considerations in mind, we first describe the attacks in MFNC systems and then we propose NCShield for defending these attacks.

A. Attack Models

ENC systems have been found to suffer from a low estimation accuracy when part of its participants lie about their coordinates, which may be caused by one of the following attacks [16]: disorder, repulsion, and isolation. Likewise, malicious nodes in MFNC systems may also falsify coordinates and/or intensively delay RTT probing packets, which we analyze and model as follows.
Repulsion attack: Malicious nodes in MFNC systems attempt to influence the victims' incoming vectors Y, deceiving other nodes to look like far away from the victims to reduce their attractiveness, $\vec{Y_{target}} = \alpha * \vec{R_1} * Y_{max} + \beta * Y_{max}$. $\vec{R_1}$ is a random-generated d-dimensional vector, in which every element is randomly choose between (0,1]. Y_{max} is the multiplier controlling the coordinate space. $\alpha + \beta = 1$ so that the target incoming vector is within a ring of coordinate

space. Malicious nodes falsify their coordinates by randomly setting within a smaller domain and try to delay RTT probes corresponding to this deceit, $\vec{X_{mal}} = \delta * \vec{R_2} * X_{max}$ and $d_{target} = \vec{X_{mal}} \cdot \vec{Y_{target}}$.

Isolation attack: The outgoing vectors of the victims become the attack targets $\vec{X_{target}} = \vec{C}$, where $\vec{C}$ is a certain d-dimensional coordinate), so that the victims believe they are near some particular nodes and try to connect to them when they need P2P neighbors. $d_{target} = \vec{X_{target}} \cdot \vec{Y_{mal}}$.

Disorder attack: There is no specific difference between the two types of vectors when attack is conducted. Normally both of them are targets. Malicious nodes falsify their both coordinates and randomly delay the RTT probing packets, $d_{target} = d_{original} + d_{delay}$. Therefore, malicious information can be injected into system, aiming at creating chaos in the system.

B. Overview of Defense Model

We employ a score and vote based approach to defend decentralized MFNC systems against the above-mentioned attacks. Extra certain nodes are chosen to score and vote when a node updates coordinates, while extra coordinate and RTT information are required for detecting the attackers. With the help of Byzantine resilient random node sampling [4] (a secure gossip-based membership protocol), each node can sample a small set of nodes in a resilient unbiased manner, so-called VList, for verifying the correctness of its neighbor nodes. Upon collecting independent opinions of all VList nodes, a final suggestion of potential malicious nodes will be reached by summing them up.

Algorithm 1 presents the pseudocode of NCShield. It consists of two main parts. One is the construction and maintenance of verification list (VList), and another is the verification process embedded in NC algorithm with the help of VList. Here, we carefully make a tradeoff between performance (high accuracy) and overhead (introducing as few new entities as possible) to avoid new vulnerabilities while ensuring efficiency.

Algorithm 1 Pseudo Code of NCShield

```
define ST SCORE_THRESHOLD
define VT VOTE_THRESHOLD
define NT VALID_NEIGHBORS_THRESHOLD
Get_Initial_Host_Candidates(RP)
Generate_NList_and_VList()
Connect_to_NList_Members()
Connect_to_VList_Members()
while forever do
    Get(d(·), X, Y)
    Check_and_Renew_Candidates_with_Brahms()
    Deliver_to_VList(X, Y, neighbor_addrs)
    for all Members ∈ VList do
        d_vton(·) = Measure_to_Neighbor(neighbor_addrs)
        score(·) = Cal_Score(d_vton(·), X, Y, X_v, Y_v)
        vote(·) = Cal_Vote(score(·), ST)
    end for
    Deliver_to_Host(vote(·))
    valid_neighbors = Parse_Vote(vote(·), VT)
    if valid_neighbors ≥ NT then
        MFNC_Update_Coordinate()
    end if
    Wait(NC_UPDATE_INTERVAL)
end while
```

C. Construction and Maintenance of VList

To implement this approach in real systems, the construction and maintenance of VList are critical. Like the NList in P2P systems, VList also consists of unbiased sampling nodes from the systems. Therefore, VList can be constructed and maintained together with NList of each node. A candidate list should also be constructed and maintained in case of invalidation of entries on NList and VList.

In this paper, we employ Brahms [4] for node list construction and maintenance. Specifically, when a node H_{new} joins the system, in addition to the necessary components required by NC system and gossip protocol, a sampler component is also planted in node H_{new} and a balance algorithm controlling the contribution of pushes and pulls in gossip session is implemented. Therefore, unbiased samplings of both NList and VList members can be achieved. More specifically, upon node H_{new} joining the system, it will operate in the following steps:

Contacting the Rendezvous Point (RP) Node H_{new} registers to the RP, and obtains the initial host candidates.

Contacting candidates Node H_{new} sends messages to nodes on the candidate list. When node H_{new} receives a response from a node, it will be added to node H_{new}'s NList or VList. Repeat this operation until both lists reach the preset scales. Thus initial sampling lists are also constructed.

Secure gossip process When node H_{new} needs to discover more nodes, it starts a Brahms process, in which, the number of nodes discovered by pushes and pulls are balanced. Along with the sample list, an unbiased new sample can be generated. Finally new unbiased sample of candidates and sample list are obtained. With the help of Brahms, malicious nodes are not able to intensively introduce other malicious nodes to victims during gossip for increasing the percentage of malicious nodes in neighbor list (NList).

D. Coordinates Verification Process

In this process, considering the distance variations over time, the trust and reputation system should be tolerant for such dynamics without treating them as malicious. We adopt the score computation as in Eq. 6 and Eq. 7, and set reasonable threshold to distinguish malicious distance alteration from distance variation.

Table I
COMPARISON OF COMMUNICATION OVERHEAD BETWEEN VERACITY AND NCSHIELD

Step No.	Veracity using DHT	Overhead	NCShield using Gossip protocol	Overhead
1	Publishers contact VSets to deliver messages	$N*u*log_2N$	H contacts NList for NList's coordinates	$N*m*2$
2	VSets ping publishers for RTTs	$N*u*2$	H sends NList's coordinates to VList	$N*u$
3	Investigator contacts publishers for coordinates	$N*m*2$	VList contact NList for coordinates	$N*m*u*2$
4	Investigator contacts VSets for evidences	$N*m*u*log_2N$	VList ping NList for RTTs	$N*m*u*2$
5	VSets return results to investigator	$N*m*u$	VList return results to H	$N*u$

Once a node H has obtained its NList and VList, it starts to update its incoming vector and outgoing vector periodically. Let m and u present the number of neighbors assigned to node H and the number of verification nodes, respectively.

(a) Node H asks its neighbor node H_n for its coordinates.
(b) Node H_n may respond with its falsified coordinates.
(c) Node H then asks its VList members, H_v nodes for help.
(d) These VList members retrieve coordinates of node H_n and conduct independent measurements to it. They calculate two scores, $score^{in}_{H_vH_n}$ and $score^{out}_{H_vH_n}$, based on the suspicious outgoing and incoming vectors of node H_n and the RTTs between node H_v and node H_n ($D(H_v, H_n)$ and $D(H_n, H_v)$), as in Eq. 6 and Eq. 7.

$$score^{in}_{H_vH_n} = \frac{|D^E(H_v, H_n) - D(H_v, H_n)|}{\min(D^E(H_v, H_n), D(H_v, H_n))} \quad (6)$$

$$score^{out}_{H_vH_n} = \frac{|D^E(H_n, H_v) - D(H_n, H_v)|}{\min(D^E(H_n, H_v), D(H_n, H_v))} \quad (7)$$

(e) Node H_v evaluates the scores (Eq. 8 and Eq. 9) according to a pre-defined score threshold (ST) and return $vote^{in}_{H_n}$ and $vote^{out}_{H_n}$ for node H_n as well as H_n's coordinates they obtained to node H.

$$vote^{in}_{H_n} = \sum_{H_v \in VList}^{v} (score^{in}_{H_vH_n} \geq ST) \quad (8)$$

$$vote^{out}_{H_n} = \sum_{H_v \in VList}^{v} (score^{out}_{H_vH_n} \geq ST) \quad (9)$$

(f) Node H integrates the returned information and calculates $vote_{H_n}$ (Eq. 10) to decide whether adopting or discarding node H_n's coordinates according to a pre-defined vote threshold (VT). Finally, after collecting vote information of all neighbors, node H calculates $valid\_neighbors$ (Eq. 11) and decides whether to start the NC update process, according to a pre-defined valid neighbors threshold (NT). Thus, an update round along with verification is completed.

$$vote_{H_n} = (vote^{in}_{H_n} \geq VT) \&\& (vote^{out}_{H_n} \geq VT) \quad (10)$$

$$valid\_neighbors = \sum_{H_n \in NList}^{m} vote_{H_n} \quad (11)$$

Besides the process described above, several details are noteworthy:

1) Before calculating $vote_{H_n}$, a comparison should be made in case that node H_n sends different coordinates to node H and its VList members. If a false comparison result appears, the corresponding $vote^{in}_{H_n}$ and $vote^{in}_{H_n}$ will be set to 0 directly.
2) The verification contacts should have no difference from the contacts of retrieving update information, to prevent node H_n from noticing the verification contacts, after which it may send the unfalsified coordinates to H_v, and stop delaying the RTT measurement requests.
3) There are various strategies of making a vote, since there are two coordinates for each node and at least one coordinate need to be verified. In this paper, we apply a relatively stringent strategy that a positive vote is made only if both of the vote results satisfy the threshold (VT).
4) Malicious nodes may also appear in VList. Brahms guarantees the unbiased percentage of malicious nodes (the same ratio of malicious nodes as that in the whole network) in VList, as well as in NList. In our simulation, they will vote randomly for NList to node H_n.

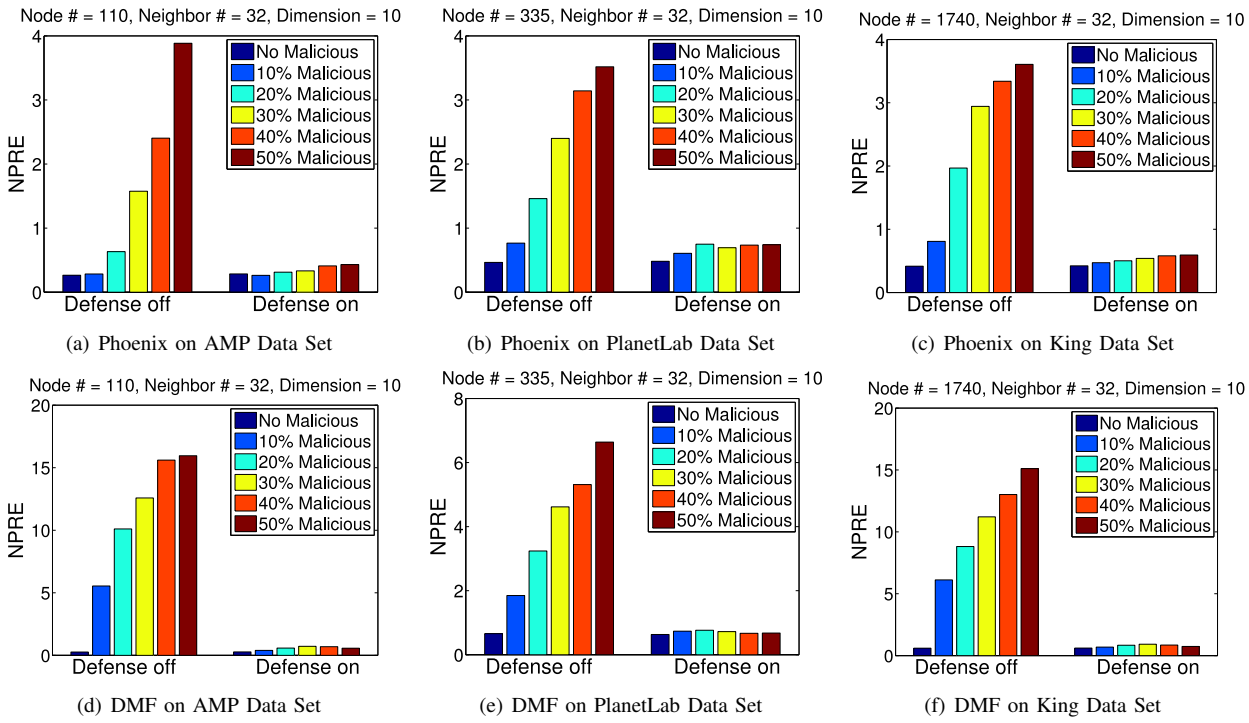
E. Communication Overhead Analysis

Table I summarizes the coordinate verification steps of Veracity using DHT and NCShield using gossip protocol. The communication overhead of each step is also listed, counted in number of messages. In the table, m and u are defined in section III-C, while N is the number of nodes in the NC system. According to the table, the gossip-based mechanism can significantly save communication costs compared with DHT-based mechanism, not to mention if the neighbor selection approach applied in an NC system is incompatible with DHT.

For typical configuration of NC systems, $m = 32$ [9] and $u = 7$ [29], with a node scale of 1024, the communication overheads in one round of coordinate verification process of all nodes are 2674688 messages and 997376 messages in Veracity and NCShield, respectively. Compared with DHT based solution, our gossip-based approach can save 62.7% traffics for coordinate verification operations.

IV. EVALUATION FOR AGGREGATE DATA SETS

In this section, we present the results of simulation study in Phoenix and DMF on aggregate data sets and dynamic data set. Simulations for each scenario are repeated 5 times and the results are the average values. We set the malicious group size from 10% to 50%, with the interval of 10%.



(a) Phoenix on AMP Data Set (b) Phoenix on PlanetLab Data Set (c) Phoenix on King Data Set

(d) DMF on AMP Data Set (e) DMF on PlanetLab Data Set (f) DMF on King Data Set

Figure 1. Disorder Attack and Defense in Phoenix and DMF on Aggregate Data Sets

Table II
NPRE OF REPULSION ATTACK AND DEFENSE

NC	Data	Defense	Percentage of Malicious Nodes			
			0%	10%	30%	50%
Phoenix	AMP	OFF	0.284	0.505	1.059	2.706
		ON	0.285	0.289	0.307	0.394
	PL	OFF	0.444	0.701	1.410	2.259
		ON	0.492	0.558	0.674	0.752
	King	OFF	0.450	1.170	1.558	4.029
		ON	0.456	0.548	0.590	0.619
DMF	AMP	OFF	0.234	3.644	6.331	8.482
		ON	0.220	0.212	0.224	0.237
	PL	OFF	0.668	4.603	14.354	22.191
		ON	0.657	0.780	0.644	0.641
	King	OFF	0.611	13.585	35.903	50.113
		ON	0.614	0.613	0.614	0.610

Table III
NPRE OF ISOLATION ATTACK AND DEFENSE

NC	Data	Defense	Percentage of Malicious Nodes			
			0%	10%	30%	50%
Phoenix	AMP	OFF	0.285	0.558	1.157	3.412
		ON	0.285	0.301	0.305	0.373
	PL	OFF	0.445	0.689	1.328	2.567
		ON	0.469	0.529	0.653	0.676
	King	OFF	0.444	0.988	1.582	5.001
		ON	0.463	0.531	0.557	0.586
DMF	AMP	OFF	0.284	2.097	5.461	7.890
		ON	0.269	0.278	0.262	0.261
	PL	OFF	0.676	1.371	2.716	3.296
		ON	0.656	0.781	0.664	0.654
	King	OFF	0.657	3.970	8.430	15.431
		ON	0.513	0.516	0.523	0.524

The main metric used in our performance evaluation is *ninetieth percentile relative error (NPRE)*, which has been widely used in [8]–[10], [24]–[26] since it guarantees 90% of the links have lower RE values than it. Smaller NPRE value indicates higher overall prediction accuracy [17], [27], [29], [34].

A. Simulation Setup for Aggregate Data Sets

In both Phoenix and DMF, each node is assigned 32 [9], [21] neighbors and 7 [29] VList members, which are selected at random. The coordinate dimension is set to 10 [9], [21]. For Phoenix, the constant C is set to 10 [9], ST (SCORE_THRESHOLD defined in Algorithm 1 in Section III-A) is 1.0, VT (VOTE_THRESHOLD) is 6, and NT (VALID_NEIGHBORS_THRESHOLD) is 10. While in DMF, the regulation coefficient λ is set to 50 [21], ST is 0.4, VT is 4, and NT is 16. Later we will see that the parameters of NCShield are data-independent.

The attackers are considered to be injected to systems successfully. This is because "malicious" is a relative conception that these attackers may act as an ordinary node in the beginning, then enable their malicious behaviors later. Using default values in [9], [21], we set update rounds of Phoenix and DMF as 30 and 50, respectively. We use three representative aggregate Internet data sets for our evaluation. The first data set is the AMP data set [24], which includes the RTTs among 110 Internet hosts. The hosts are mainly at

NSF supported HPC sites, with about 10% outside the US. AMP data set has been used in [24], [36]. The second data set is the PlanetLab data set [39], which includes the RTTs among 335 PlanetLab hosts all over the world, collected during March-April, 2010. PlanetLab data set has been used in [13], [24], [36], [39]. The third data set is King data set which includes the RTTs among 1740 Internet DNS servers [13]. King data set has been used in [13], [31], [32], [36]. These data sets can present three different Internet delay spaces [35].

B. Effect of Our Defense on Aggregate Data Sets

As described above, in disorder attack, malicious nodes would send false coordinates, both outgoing vectors and incoming vectors, which are randomized within the maximum value of each type of vectors. In addition, the malicious nodes intensively delay the probes with the range of [100..1000] ms. Fig. 1 shows the results of disorder attack and defense simulation. The NPRE results are calculated with all the participant nodes *EXCEPT* malicious nodes. The figure indicates that NCShield can achieve significant defense performance towards disorder attack. With our defense approach, the NPRE of each experiment with different percentage of malicious nodes do not increase remarkably. For example, in the simulation of Phoenix on AMP data set (Fig. 1(a)), the NPRE of normal system is 0.223. When 30% malicious nodes exist, the NPRE turns to be 0.998. The RE increases significantly, indicating the accuracy of Phoenix is suffering from great degradation. While NCShield is activated, the NPRE drops to 0.315, indicating prediction accuracy degradation is suppressed remarkably. Fig. 1(b)-1(f), describing the results of Phoenix on PL335 and King data sets, DMF on AMP, PL335 and King data sets, respectively, also show the similar defense performance of NCShield. As the percentage of malicious nodes increases, the performance of the system gets significantly worse.

In repulsion attack, both the coordinates of malicious nodes are randomized within a half of the maximum of outgoing vector and incoming vector, respectively. The incoming vectors of victims are attacked to be set within 0.7 to 1.0 maximum coordinates, randomly. Thus, the RTT probes are intensively delayed to correspond with target coordinates. Table II shows the simulation results of repulsion attack and defense on both systems and on all three aggregate data sets, presented in NPRE. The NPRE results are calculated with the outgoing vectors of all nodes *EXCEPT* malicious nodes and the incoming vectors of victim nodes. From the table we can see that our defense approach can significantly protect victims under repulsion attacks. That means the influence aiming at the incoming vectors of victims can be identified and terminated. Therefore, other nodes would not believe the victims are far away from them.

In isolation attack, the coordinates of malicious are randomized the same way as in repulsion attack. However, the attack targets are outgoing vectors of victims, which are aimed to be set to maximum coordinates. The RTT probes are also intensively delayed correspondingly. Table III shows the simulation results of isolation attack and defense on both systems and on all three aggregate data sets, presented in NPRE. The NPRE results are calculated with the outgoing vectors of victim nodes and the incoming vectors of all nodes *EXCEPT* malicious nodes. From the table we can see that our defense approach can significantly protect victims under isolation attacks, reducing the attack influence to acceptable level. Thus, the victims are no longer convinced to be in a remote field of the coordinate space.

Table IV
AVERAGE NPRE OF SIMULATION ON "K200-ALLPAIRS-1H" DATA SET

NC	Type	Defense	Percentage of Malicious Nodes			
			0%	10%	30%	50%
Phoenix	Dis.	OFF	0.500	0.878	2.790	5.416
		ON	0.501	0.612	0.703	1.013
	Rep.	OFF	0.500	0.903	1.828	3.372
		ON	0.500	0.530	1.078	1.157
	Iso.	OFF	0.500	2.362	1.606	4.510
		ON	0.500	0.525	1.066	1.180
DMF	Dis.	OFF	0.924	1.684	3.218	4.012
		ON	0.858	0.938	1.344	1.613
	Rep.	OFF	0.924	11.031	18.116	22.660
		ON	0.858	1.990	0.798	0.899
	Iso.	OFF	0.924	11.471	20.191	25.920
		ON	0.858	1.643	0.855	0.984

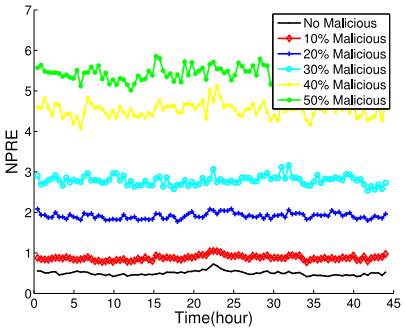
For simulation results of every data set, Phoenix shows better performance than DMF under different attacks. The weight mechanism seems to play a critical role, so that Phoenix is resilient to small amount ($< 10\%$) of malicious nodes.

V. EVALUATION FOR DYNAMIC DATA SET AND ONLINE GAME SCENARIO

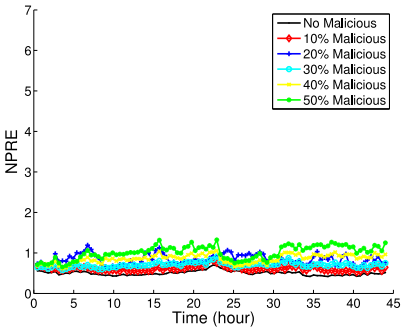
A. Simulation Setup for Dynamic Data Set

The real Internet distances are time varying, which is especially critical in network monitoring scenario. However, existing security schemes for ENC systems do not consider this variation at all. A practical security scheme should be able to distinguish between ordinary distance variation and maliciously generated distance providing by the attackers. We evaluate NCShield on the "K200-allpairs-1h" dynamic data set [22]. This data set contains 200 nodes and the data collection lasts 44 hours using King method. We have obtained 99 continuous snapshots of all pairwise RTTs.

In this simulation, we use the same parameter settings of Phoenix, DMF and NCShield defined in Section IV-A. We run on the first one of 99 snapshot matrices to achieve an acceptable convergence of coordinates. Then from the second snapshot matrix, the malicious nodes start their attacks.



(a) Phoenix w/o Defense



(b) Phoenix with Defense

Figure 2. Disorder Attack in Phoenix on "K200-allpairs-1h" Data Set

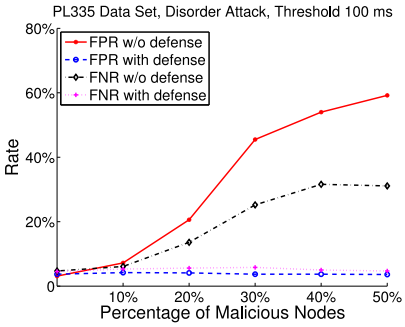


Figure 3. Online Game Scenario Evaluation

B. Effect of Our Defense on Dynamic Data Set

In this simulation, malicious nodes perform the same behaviors of disorder, repulsion and isolation attacks as those in Section IV-B.

Fig. 2 shows the NPRE variations in Phoenix under disorder attack on "K200-allpairs-1h" data set. In Fig. 2(a), without defense, the RE increases significantly when malicious nodes increase. As Internet distances varying from time to time, the performance degrades when the system is under disorder attack. While in Fig. 2(b), with NCShield, the increasing of RE is obviously mitigated, which indicates NCShield works well in this scenario.

Table IV lists the average NPREs of all three scenarios both in Phoenix and DMF systems. From the table we can see that NCShield can remedy the three attacks significantly for both systems. For example, in Phoenix with 30% malicious nodes, NCShield can help decrease the average NPRE from 1.073 to 0.717 under disorder attack. In DMF under 10% malicious node conducting repulsion attack, NCShield decreases the average NPRE from 8.796 to 0.772.

C. Online Game Scenario Evaluation

Besides using typical RE metric for evaluating the prediction accuracy, we also evaluate NCSheild in a practical scenario, i.e., a popular online game scenario. As introduced in [6], [23], identifying all end-to-end links with shorter latencies than a predefined threshold is critical for various real-time interaction games. According to the requirement of first-person perspective games, we set this threshold as 100ms [6]. A link is defined as a good (resp. bad) link when its measured RTT is below (resp. above) the predefined threshold. We use Phoenix for link classification according to predicted distances. A true positive (TP) indicates that a good link is correctly predicted as "good", while a false positive (FP) shows that a bad link is wrongly predicted as "good". Likewise, a true negative (TN) tells that a bad link is correctly predicted as "bad", while A false negative (FN) points that a good link is wrongly predicted as "bad". False positive rate (FPR) and false negative rate (FNR) are defined by $FPR = FP/(FP + TN)$ and $FNR = FN/(TP + FN)$, respectively.

We conduct this simulation on AMP, PL335 and King data set. All parameters of Phoenix and NCShield are the same as those in section IV. Half of the nodes, i.e. 55 nodes, 167 nodes and 870 nodes, which are always victims as malicious percentage increases, are chosen to be participants of online game. The threshold is set to 100 ms.

Due to page limitation, we only present the result of PL335 data set in Fig. 3, while results of the other two data sets are similar. When 30% malicious node conducting disorder attack, NCShield can reduce the FPR and FNR from 45.5% and 25.2% to 3.7% and 5.8%, respectively. Most of the negative impacts on link selection introduced by attackers are eliminated. This shows that in an online game scenario using Phoenix for link selection, NCShield is practical to prevent performance degradation caused by disorder attacks.

VI. Conclusions

As decentralized MFNC systems can scale to millions of users and is more realistic than Euclidean-based NC systems by not assuming the triangle inequality, they have become more suitable to large-scale distributed applications on the Internet. However, coming with the benefits are also vulnerabilities that must be addressed. As we have shown in this paper with the disorder, repulsion and isolation attacks toward Phoenix and DMF systems, the security threats of decentralized MFNC systems can be very severe, and even a small number of malicious nodes can deteriorate the accuracy of an MFNC system significantly.

We proposed a score and vote based approach with an effective and scalable node sampling mechanism. Through simulations using both aggregate data sets and a dynamic data set, our approach is able to effectively defend MFNC systems from attacks. In particular, our approach differs from previous approaches and is able to distinguish between legitimate distance variations and malicious distance alterations.

We also experimented how an application may use a MFNC system protected with our approach. We designed an online game link selection experiment based on Phoenix, and showed that our approach can significantly reduce the negative effects and ensure the quality of service. Our future work in this regard includes the investigation of the new emerging frog-boiling attacks [7], the development of a more practical application, and the evaluation with both Phoenix and DMF systems on a real network.

ACKNOWLEDGMENT

The authors are grateful to Xiaohan Zhao from University of California, Santa Barbara, for helpful suggestions. We also thank the anonymous reviewers for the valuable comments and suggestions.

REFERENCES

[1] I. Abraham, D. Malkhi. Compact Routing on Euclidian Metrics. In Proc. of ACM PODC, 2004.
[2] S. Agarwal, J. Lorch. Matchmaking for Online Games and Other Latency-Sensitive P2P Systems. In Proc. of ACM SIGCOMM, 2009.
[3] H. Attiya and J. Welch. Distributed Computing: Fundamentals, Simulations, and Advanced Topics. John Wiley and Sons, Inc., 2004.
[4] E. Bortnikov, M. Gurevich, *et al.* Brahms: Byzantine Resilient Random Membership Sampling. In Proc. of ACM PODC, 2008.
[5] M. Castro, P. Drushel, *et al.* Secure Routing for Structured Peer-to-Peer Overlay Networks. In Proc. of OSDI, 2002.
[6] M. Claypool, K. Claypool. Latency and player actions in online games. Communications of the ACM (2006) Volume 49, Issue 11.
[7] E. Chan-Tin, V. Heorhiadi, *et al.* The Frog-Boiling Attack: Limitations of Secure Network Coordinate Systems. ACM Transactions on Information and Systems Security (TISSEC), 14(3):1-28, 2011.
[8] Y. Chen, P. Sun, X. Fu, T. Xu. Improving Prediction Accuracy of Matrix Factorization Based Network Coordinate Systems. In Proc. of IEEE ICCCN, 2010.
[9] Y. Chen, X. Wang, *et al.* Phoenix: A Weight-based Network Coordinate System Using Matrix Factorization. IEEE Trans. on Network and Service Management, 8(4):334-347, 2011.
[10] Z. Chen, Y. Chen, Y. Zhu, *et al.* Tarantula: Towards an Accurate Network Coordinate System by Handling Major Portion of TIVs. In Proc. of IEEE GLOBECOM, 2011.
[11] Z. Chen, Y. Chen, C. Ding, *et al.* Pomelo: Accurate and Decentralized Shortest-path Distance Prediction in Social Graphs. ACM SIGCOMM Computer Communication Review, 41(4):406-407, 2011.
[12] M. Costa, M. Castro, A. Rowstron, *et al.* PIC: Practical Internet Coordinates for Distance Estimation. In Proc. of IEEE ICDCS, 2004.
[13] F. Dabek, R. Cox, and F. Kaashoek. Vivaldi: A Decentralized Network Coordinate System. In Proc. of ACM SIGCOMM, 2004.
[14] C. Ding, Y. Chen, T. Xu, X. Fu. CloudGPS: A Scalable and ISP-Friendly Server Selection Scheme in Cloud Computing Environments. In Proc. of 20th IEEE/ACM IWQoS, 2012.
[15] A.J. Ganesh, A.-M. Kermarrec, *et al.* Peer-to-Peer Membership Management for Gossip-Based Protocols. IEEE Trans. on Computers, 52(2): 139–149, 2003.
[16] M. A. Kaafar, L. Mathy, *et al.* Real Attacks on Virtual Networks: Vivaldi out of Tune. In Proc. of SIGCOMM Workshop on Large-Scale Attack Defense, 2006.
[17] M. A. Kaafar, L. Mathy, *et al.* Securing Internet Coordinate Embedding Systems. In Proc. of ACM SIGCOMM, 2007.
[18] J. Ledlie, P. Gardner, and M. Seltzer. Network Coordinates in the Wild. In Proc. of NSDI, 2007.
[19] J. Ledlie, P. Pietzuch, and M. Seltzer, Stable and Accurate Network Coordinates. In Proc. of IEEE ICDCS, 2006.
[20] S. Lee, Z. Zhang, *et al.* On Suitability of Euclidean Embedding of Internet Hosts. In Proc. of ACM SIGMetrics/Performance, 2006.
[21] Y. Liao, P. Geurts, *et al.* Network Distance Prediction Based on Decentralized Matrix Factorization. In Proc. of IFIP-TC6 Networking, 2010.
[22] C. Lumezanu, R. Baden, N. Spring, *et al.* Triangle Inequality Variations in the Internet. In Proc. of ACM IMC, 2009.
[23] J. Manweiler, S. Agrawal, *et al.* SwitchBoard: A Matchmaking System for Multiplayer Mobile Games. In Proc. of ACM Mobisys, 2011.
[24] Y. Mao, L. Saul, J. M. Smith. IDES: An Internet Distance Estimation Service for Large Network. IEEE Journal on Selected Areas in Communications, 24(12): 2273–2284, 2006.
[25] T.S.E. Ng, H. Zhang. Predicting Internet Network Distance with Coordinates-based Approaches. In Proc. of IEEE INFOCOM, 2002.
[26] T.S.E. Ng, H. Zhang. A Network Positioning System for the Internet. In Proc. of USENIX ATC 2004.
[27] D. Saucez, B. Donnet, and O. Bonaventure. A reputation-based approach for securing vivaldi embedding system. In Dependable and Adaptable Networks and Services, 2007.
[28] P. Sharma, Z. Xu, S. Banerjee, and S. Lee. Estimating network proximity and latency. In ACM SIGCOMM CCR, 36(3): 39-50, 2006.
[29] M. Sherr, M. Blaze, *et al.* Veracity: Practical Secure Network Coordinates via Vote-based Agreements. In Proc. of USENIX ATC, 2009.
[30] M. Steiner and Ernst W. Biersack. Where is my Peer? Evaluation of the Vivaldi Network Coordinate System in Azureus. In Proc. of 8th International IFIP-TC6 Networking, 2009.
[31] G. Wang, B. Zhang, T.S.E. Ng. Towards Network Triangle Inequality Violation Aware Distributed Systems. In Proc. of ACM IMC, 2007.
[32] G. Wang, T.S.E. Ng. Distributed Algorithms for Stable and Secure Network Coordinates. In Proc. of ACM IMC, 2008.
[33] P. Wendell, J. W. Jiang, *et al.* DONAR: Decentralized Server Selection for Cloud Services. In Proc. of ACM SIGCOMM, 2010.
[34] D. Zage and C. Nita-Rotaru. On the accuracy of decentralized virtual coordinate systems in adversarial networks. In Proc. of ACM CCS, 2007.
[35] B. Zhang, T.S.E. Ng, *et al.* Measurement-Based Analysis, Modeling, and Synthesis of the Internet Delay Space. In Proc. of ACM IMC, 2006.
[36] R. Zhang, C. Tang, *et al.* Impact of the Inaccuracy of Distance Prediction Algorithms on Internet Applications: an Analytical and Comparative Study. In Proc. of IEEE INFOCOM, 2006.
[37] X. Zhao, A. Sala, H. Zheng, Ben Y. Zhao. Efficient Shortest Paths on Massive Social Graphs. In Proc. of CollaborateCom, 2011.
[38] H. Zheng, E.K. Lua, M. Pias, *et al.* Internet Routing Policies and Round-Trip-Times. In Proc. of PAM, 2005.
[39] Y. Zhu, Y. Chen, *et al.* Taming the Triangle Inequality Violations with Network Coordinate System on Real Internet. In Proc. of ACM ReArch, 2010.