Securiting Internet coordinates systems

Mohamed Ali Kaafar

Université de Nice - Sophia Antipolis
École Doctorale STIC

THÈSE

Présentée pour obtenir le titre de :

Docteur en Sciences de l'Université de Nice - Sophia Antipolis

Spécialité : INFORMATIQUE

par

Mohamed Ali KAAFAR

Équipe d'accueil : Planète – INRIA Sophia Antipolis

SECURING INTERNET COORDINATES SYSTEMS

Soutenue publiquement à l'INRIA le 21 Septembre 2007 devant le jury composé de :

Président :	Jean-Paul	RIGAULT	University of Nice Sophia Antipolis, France
Rapporteurs :	Jon	CROWCROFT	University of Cambridge, UK
	Serge	FDIDA	University of Paris 6, France
Examinateurs :	Katia	OBRACZKA	University of California Santa Cruz, USA
	Laurent	MATHY	Lancaster University, UK
	Farouk	KAMOUN	University of Manouba, TN
Co-directeurs :	Thierry	TURLETTI	INRIA, France
	Walid	DABBOUS	INRIA, France

LA SÉCURITÉ DES SYSTÈMES DE COORDONNÉES INTERNET

SECURING INTERNET COORDINATES SYSTEMS

MOHAMED ALI KAAFAR
September 2007

To the best paint life ever made, my Cyrina, my Chocolate

To my beloved mother. I owe to you all what I am. Your prayers always followed me.

To the one who is always in my mind, my beloved father Meni3

SECURING INTERNET COORDINATES SYSTEMS

Mohamed Ali Kaafar

ABSTRACT

Internet coordinate-based systems allow easy network positioning. In such systems, the basic idea is that if network distances between Internet nodes can be embedded in an appropriate space, unmeasured distances can be estimated using a simple distance computation in that space. Recently, these coordinates-based systems have been shown to be accurate, with very low distance prediction error. However, most, if not all, of current proposals for coordinate systems assume that the nodes partaking in the system cooperate fully and honestly with each other – that is that the information reported by probed nodes is correct – this could also make them quite vulnerable to malicious attacks. In particular, insider attacks executed by (potentially colluding) legitimate users or nodes infiltrating the system could prove very effective.

As the use of overlays and applications relying on coordinates increases, one could imagine the release of worms and other malware, exploiting such cooperation, which could seriously disrupt the operations of these systems and therefore the virtual networks and applications relying on them for distance measurements.

In this thesis, we first identify such attacks, and through a simulation study, we observed their impact on two recently proposed positioning systems, namely Vivaldi and NPS. We experimented with attack strategies, carried out by malicious nodes that provide biased coordinates information and delay measurement probes, and that aim to (i) introduce disorder in the system, (ii) fool honest nodes to move far away from their correct positions and (iii) isolate particular target nodes in the system through collusion. Our findings confirm the susceptibility of the coordinate systems to such attacks.

Our major contribution is therefore a model for malicious behavior detection during coordinates embedding. We first show that the dynamics of a node, in a coordinate system without abnormal or malicious behavior, can be modeled by a Linear State Space model and tracked by a Kalman filter. Then we show, that the obtained model can be generalized in the sense that the parameters of a filter calibrated at a node can be used effectively to model and predict the dynamic behavior at another node, as long as the two nodes are not too far apart in the network. This leads to the proposal of a Surveyor infrastructure: Surveyor nodes are trusted, honest nodes that use each other exclusively to position themselves in the coordinate space, and are therefore immune to malicious behavior in the system. During their own coordinate embedding, other nodes can then use the filter parameters of a nearby Surveyor as a representation of normal, clean system behavior to detect and filter out abnormal or malicious activity. A combination of simulations and PlanetLab experiments are used to demonstrate the validity, generality, and effectiveness of the proposed approach for both Vivaldi and NPS.

Finally, we address the issue of asserting the accuracy of Internet coordinates advertised by nodes of Internet coordinate systems during distance estimations. Indeed, some nodes may even lie deliberately about their coordinates to mount various attacks against applications and overlays.

Our proposed method consists in two steps: 1) establish the correctness of a node's claimed coordinate by using the Surveyor infrastructure and malicious embedding neighbor detection; and 2) issue a time limited validity certificate for each verified coordinate. Validity periods are computed based on an analysis of coordinate inter-shift times observed by Surveyors. By doing this, each surveyor can estimate the time until the next shift and thus, can limit the validity of the certificate it issues to regular nodes for their calculated coordinates. Our method is illustrated using a trace collected from a Vivaldi system deployed on PlanetLab, where inter-shift times are shown to follow long-tail distribution (log-normal distribution in most cases, or Weibull distribution otherwise). We show the effectiveness of our method by measuring the impact of a variety of attacks, experimented on PlanetLab, on distance estimates.

La sécurité des systèmes de coordonnées Internet

Mohamed Ali Kaafar

Résumé

Les systèmes Internet à base de coordonnées permettent un positionnement pratique des nœuds dans le réseau. Dans ce type de systèmes, l'idée principale est que si les distances réseau entre différents nœuds Internet peuvent être plongées dans un espace approprié, alors les distances non mesurées peuvent être estimées en utilisant une simple opération de calcul de distance géométrique dans cet espace. Récemment, on a pu prouver que ces systèmes à base de coordonnées étaient précis, avec une faible erreur de prédiction. Cependant, ces systèmes se basent souvent sur une coordination entre les nœuds, et font l'hypothèse que les informations reportées par les nœuds sont correctes.

Dans cette thèse, nous avons identifié plusieurs attaques, exploitant cette hypothèse d'honnêteté des nœuds, et pouvant être lancées contre des systèmes de positionnement Internet à base de coordonnées. Nous avons en l'occurrence étudié l'impact qu' avaient de telles attaques sur deux systèmes représentatifs des systèmes de positionnement actuels : NPS et Vivaldi. Nous avons entre autres montré que ces attaques, pouvaient dangereusement mettre en péril le bon fonctionnement de ces systèmes de coordonées, et par la même les applications se basant sur ce système pour les estimations de distances. À travers les simulations de plusieurs attaques, menées par des nœuds malhonnêtes, fournissant des coordonnées biaisées ou retardant les mesures, nous avons expérimenté plusieurs stratégies d'attaques qui ont pour objectifs: (i) d'introduire du désordre dans le système, (ii) de tromper les nœuds honnêtes afin qu'ils se positionnent loin de leurs coordonnées correctes et (iii) d'isoler certains nœuds cibles à travers des collusions. Nos résultas confirment la vulnérabilité de tels systèmes à ces attaques.

Notre contribution majeure a été par la suite de proposer un modèle de détection des comportements malicieux au sein de ces systèmes de positionnement durant le calcul des coordonnées.

Nous avons montré en premier lieu que la dynamique d'un nœud, dans un système de coordonnées, exempt de comportements anormaux ou malhonnêtes, peut être modélisée par un modèle d'états linéaire, et traqué par un filtre de Kalman. De plus, les paramètres d'un filtre calibré au niveau d'un nœud donné, peuvent être utilisés pour modéliser et prédire le comportement dynamique d'un autre nœud, tant que ces deux nœuds sont proches l'un de l'autre dans le réseau. Nous avons dès lors proposé une infrastructure de nœuds experts : des nœuds de confiance, se positionnant dans l'espace des coordonnées, en utilisant exclusivement d'autres nœuds experts. Ils sont alors immunisés contre n'importe quel comportement malicieux dans le système. Pendant le calcul de leurs propres coordonnées, les autres nœuds utilisent les paramètres du filtre d'un nœud expert proche, comme étant une représentation d'un comportement normal, pour détecter et filtrer toute activité malicieuse ou anormale. Une combinaison de simulations et d'expérimentations PlanetLab a été utilisée pour démontrer la validité, la généralité et l'efficacité de l'approche proposée pour chacun des deux systèmes Vivaldi et NPS.

Enfin, nous nous sommes penchés sur le problème de la validité des coordonnées Internet telles qu' annoncées par les nœuds d'un système de coordonnées durant la phase d'estimation des distances. En effet, certains nœuds peuvent délibérément mentir quant à la valeur exacte de leurs coordonnées afin de lancer diverses attaques contre les applications et les réseaux de couverture.

La méthode proposée se divise en deux étapes : 1)établir l'exactitude des coordonnées annoncées en utilisant l'infrastructure des nœuds experts et la méthode de détection des nœuds malicieux, et 2) délivrer un certificat à validité limitée pour chaque coordonnée vérifiée. Les périodes de validité sont calculées à partir d'une analyse des temps d'inter-changement observés par les nœuds experts. En faisant cela, chaque nœud expert, peut estimer le temps jusqu'au prochain changement de coordonnées, et ainsi, peut limiter le temps de validité du certificat qu'il délivrerait aux nœuds normaux. Notre méthode est illustrée en utilisant une trace recueillie à partir d'un système Vivaldi déployé sur PlanetLab, où les distributions de temps d'inter-changements suivent des distributions longue traine (distribution log-normale dans la plupart des cas, et distribution Weilbull sinon). Nous montrons l'efficacité de notre méthode en mesurant l'impact de plusieurs attaques sur les estimations de distance, expérimentées sur PlanetLab.

ACKNOWLEDGMENTS

The research that has gone into this thesis has been thoroughly enjoyable. That enjoyment is largely a result of the interaction that I have had with my supervisors, colleagues and INRIA's people.

I feel very privileged to have worked at INRIA with my supervisors, Walid Dabbous and Thierry Turletti. To each of them I owe a great debt of gratitude for their patience, inspiration and friendship. Sincere thanks to both of them for having always been there to listen to and comment upon whatever I had thought out or wanted to discuss about and patiently reviewing all my work. Their enthusiastic nature helped me perform better. They had taught me a great deal about the field of Computer Science by sharing with me the joy of discovery and investigation that is the heart of research.

This dissertation would not have been possible without many collaborative efforts with Laurent Mathy, Chadi barakat and Kavé Salamatian. They demonstrated hard work, clear thinking, and remarkable writing skills that I learned through collaboration with them. I am deeply grateful to Laurent for his brilliant insights, thoughtful discussions, and detailed comments. I am indebted to Laurent for being of friend and for giving me invaluable advice on countless occasions not only on research but also on various issues of life. He is living evidence that even a man of genius can be warm, kind and always encouraging. I would like also to address special thanks to Chadi, for numerous technical discussions, but also for having always been a friend, that makes enjoyable working at PLANETE. Many Thanks Kavé for being the most impressive guy I've known:-). Kave's command and ease in stochastic modeling are something I aspire to achieve one day. I am also grateful to Prof. Farouk Kamoun, my master thesis' advisor at the Cristal Lab ENSI, Tunisia, who taught me never to be afraid of a challenge and always to tackle a hard problem.

Many thanks to all of these great men for being the best mentors a student can possible have. Many thanks to Nina Taft and Dina papagiannaki, shepherd of CoNext and SIGCOMM papers, providing useful reviews and comments that helped improve this dissertation.

I would like also to thank the members of my PhD committee who patiently spent their effort in reading this dissertation: Prof. Jon Crowcroft, Prof. Serge Fdida, and Dr.

Katia Obraczka. Many thanks go to Prof. Jean-Paul Rigault for serving as the chairman of the jury committee. Thanks also to our team assistant Aurelie Richard for having arranged the defense with care and for help on various matters in the past few years.

My gratitude goes also to several people, who made my stay at Sophia and Antibes easy and fun. Many thanks to the members of the squash team of INRIA, Francis, Franck, Olivier, ... I will also never forget coffee breaks, with endless philosophic discussions, fishing time and lovely moments spent with my friends Khaled, Julien J.Pascal, Mathieu, Malli, Ridha, Najeh, Nan-Ting,...

I cannot thank my parents, Bechira and Manaa, enough for their endless love, support, and encouragement. I am a proud son of such awesome parents and hope that my future baby, will feel the same way. I would also like to thank my sister, Dorsaf, for her support. Oussama, my beloved brother, come on!,... Try hard, never give up!

Finally, I am especially grateful to my wonderful wife, Cyrine, who has encouraged me so much over the years, and has been at the same time, a crew mate, and a research collaborator.

Dali,

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1

Thesis Presentation

Internet coordinate-based systems are poised to become an important service to support overlay construction and topology-aware applications. Indeed, through network distance embedding into an appropriate geometric space, such systems allow for accurate network distance estimations with low overhead. However, coordinate systems often rely on good cooperation between nodes for correct coordination and assume that information reported by probed nodes is correct.

In the first part of this thesis, we identify various attacks against coordinate embedding systems and show their effectiveness on two representative positioning systems. Our studies [1, 2] demonstrate that these attacks can seriously disrupt the operations of these systems and therefore the virtual networks and applications relying on them for distance measurements.

In a second step, we propose a general method for malicious behavior detection during coordinate computations [3]. We postulate and verify that, in the absence of malicious activity, a node's coordinate can be viewed as a stochastic process with linear dependencies whose evolution can be tracked by a Kalman filter [4]. Using secure filter parameters, nodes involved in the Internet coordinate-based system run locally and in a "stand-alone" fashion a Kalman filter tracking the coordinate adjustments. These nodes can then use the Kalman filter output (the innovation process), to compare their observed coordinate adjustments with the one predicted by the Kalman filter, and flag as "suspicious" embedding steps where the difference would be too high.

Finally, we propose to leverage the Surveyor infrastructure and embedding cheat detection test to address the question of guaranteeing the veracity of the coordinates advertised by nodes, during the distance estimation phase of these coordinate-based systems [5]. More precisely, we propose that several Surveyors measure their distance to a node in order to verify the correctness of its claimed coordinate (using the cheat detection test). If all Surveyors agree that this coordinate is the node's true coordinate, a time limited validity certificate, including the certified coordinate, is issued to the node.

1.1 Internet coordinate-based positioning systems

Recent years have seen the advent of Internet applications (e.g. PASTRY [6], OCEANSTORE [7], ESM [8], SKYPE [9], Azureus [10], etc.), which are built upon and benefit from topology-aware overlays. In particular, most if not all of these applications and associated overlays rely on the notion of network proximity, usually defined in terms of network delays or round-trip times (RTTs), for optimal neighbor selection. However, proximity measurements, based on repeated pair-wise distance measurements between nodes, can prove to be very onerous in terms of measurement overheads. Indeed, the existence of several overlays simultaneously can result in significant bandwidth consumption by proximity measurements (i.e. ping storms) carried out by individual overlay nodes [40]. Also, measuring and tracking proximity within a rapidly changing group requires high frequency measurements.

To palliate this problem, network positioning systems, such as [11, 12, 13, 14, 15, 16, 17], were introduced. The key idea of such systems is that if each node can be associated with a "virtual" coordinate in an appropriate space, distance between nodes can be trivially computed without the overhead of a direct measurement. In other words, these systems embed latency measurements amongst samples of a node population into a geometric space and associate a network coordinate vector (or coordinate in short) in this geometric space to each node, with a view to enable accurate and cheap distance (i.e. latency) predictions amongst any pair of nodes in the population. Put simply, if network distances can be embedded into a coordinate space where a reasonably accurate position for each node can be established, the measurement overhead produced by this positioning can be amortized over many (un-measured) distance predictions, which drastically reduces the distance measurement sampling cost of the overall system.

Extensive measurements and analysis from a live, large-scale deployment have shown network coordinate systems to be fit for purpose [18], making them a valuable tool to support distributed applications, systems and overlays (e.g., [19, 10, 9]) that rely on, and benefit from, the notion of network topology-awareness. In fact, coordinate-based positioning systems achieve desirable properties (as compared to complete measurement of all inter-node delays), such as accuracy, robustness, stability, scalability and low overheads. However, it should also be noted that these can often only be realized at the expense of rather slow convergence times. In such a scheme, new nodes joining the system only reach a good estimate of their own coordinates after a lapse of time in the timescale of tens of seconds to several minutes. This is several orders of magnitude slower that what is achievable with direct distance measurements between nodes and is often unacceptable for topology-aware applications whose aim is to quickly identify “best nodes”. We therefore contend that coordinate-based positioning systems are an attractive proposition only if they are deployed as a service: every node could run a coordinate system daemon at boot time which would then be capable of providing accurate coordinate estimates to applications and their overlays on request. In essence, the coordinate system could then be seen as a component of a “virtual infrastructure” that supports a wide range of overlays and applications, and this would argue in favor of a deployment of Internet coordinate systems as an always-on, large-scale service.

1.2 Motivations, Problematics and Contributions

As the use of overlays and applications relying on coordinates increases, one could imagine the release of worms and other malicious software whose purpose is to attack the coordinate system. In particular, it is very interesting to note that a system providing an “always-on and large scale coordinate service” (as an expected deployment of Internet Coordinate systems) would also likely be a prime target for hackers, as its disruption could result in the mis-functioning or the collapse of very many applications.

As current proposals for coordinate systems assume that the nodes partaking in the system cooperate fully and honestly with each other – that is that the information reported by probed nodes is correct – this make them quite vulnerable to malicious attacks. In particular, insider attacks executed by (potentially colluding) nodes infiltrating the system could prove very effective. In essence, the key questions we should ask are:

1. What if a malicious node provides biased coordinates information and/or delay measurement probes to other honest nodes in the system?

2. How one could secure the computation of coordinates in these systems in a general and effective way?

3. Even though the coordinates computation phase of these systems may have been secured, this would not prevent a malicious node from blatantly lying about its coordinate when a node requests them for simple distance estimation, during the “usage phase” of the coordinate service. How could we assert such exchanged coordinates?

Our first contribution is then to study how potent this danger is for coordinates-embedding systems. We identify three types of potential attacks against coordinate-based network positioning systems. Specifically, we study how these attacks can lead to inaccuracy of distance prediction. We demonstrate that it is easy to perform Denial of Service (DoS) attacks on such systems. We also analyze simple ways that allow malicious nodes to take control of the embedding coordinates system, as they are able to impose positions in the network to other honest nodes, without being detected. Finally, we study how conspiracy can be achieved in these systems and how much it could affect them. The “effectiveness” of these attacks on the target systems are demonstrated through extensive simulations, using two recently proposed coordinate-based positioning systems, namely Vivaldi as a prominent representative of purely peer-to-peer-based (i.e. without infrastructure support) positioning systems,and NPS as typical of landmark-based systems. This constitutes a pioneer work in the area.

We have shown that infrastructure-based systems can, under some well chosen attack strategies, be as vulnerable than those based on the peer-to-peer paradigm. Furthermore, the security mechanisms that have been proposed to date to defend against malicious nodes are clearly rather primitive and still in their infancy and definitely cannot defend against all types of attacks.

One of our salient findings is that larger systems are consistently more resilient than smaller ones. Given the observation in [15] and [14] that larger systems are more accurate and the well known fact that larger systems converge slower at start-up time, there seems to be a compelling case for large-scale coordinate systems to be built as a virtual infrastructure service component. The paradox is of course that always-on, large scale systems supporting many different applications will always attract more attacks than systems with a smaller reach, while the large size of the system itself would act as a particularly good terrain to create especially virulent propagation of the attack.

Our results also show that there is an intrinsic trade-off to be made between accuracy and vulnerability. Indeed, we have shown that the more accurate the system for a given system size, the more susceptible it was to a same proportionate level of attack.

Also, we have shown that while an attack is in full swing, the performance of the coordinate systems (and of the applications it supports) can easily degrade below that of a system where coordinates are chosen randomly, whilst the aftermath of an attack could have very long lasting effects on the system due to a small number of remaining malicious nodes.

Guided by the understanding of attack mechanisms and of their consequences on the coordinate systems gained from this study, one of our major contributions was naturally the proposal of a general and effective protocol to detect malicious behavior and to secure the embedding process in such coordinates-based systems.

We postulate and verify that, in the absence of malicious activity, a node's coordinate can be viewed as a stochastic process with linear dependencies whose evolution can be tracked by a Kalman filter [4]. Noting that, in the absence of malicious nodes, a node's coordinate depends on the combination of network conditions and the specificities of the embedding process itself (e.g. which coordinate protocol is in use, the chosen dimensionality of the geometric space, etc), we therefore introduce the concept of *Surveyor nodes* (or *Surveyors* in short). Surveyors form a group of trusted (honest) nodes, scattered across the network, which use each other *exclusively* to position themselves in the coordinate space. Of course, Surveyors do assist other nodes in their positioning (as prescribed by the embedding protocol), but we stress that Surveyors never rely on non-Surveyor nodes to compute their own coordinate. This strategy thus allows Surveyors to experience and learn the natural evolution of the coordinate space, as observed by the evolution of their own coordinate, in the absence of malicious activities. In essence, Surveyor nodes are thus vintage points guaranteed to be immune from malicious activities. The idea is that Surveyors can then share a "representation" of normal behavior in the system with other nodes to enable them to detect and filter out abnormal behavior.

Our detection protocol is then based on the fact that each Surveyor computes and calibrates the parameters of a linear state space model (in a clean-system) and then shares the parameters of this model with other nodes. These nodes can then use these parameters, to run locally and in a "stand-alone" fashion a Kalman filter tracking the coordinate adjustments. These nodes use the Kalman filter output (the innovation process), to compare their observed coordinate adjustments with the one predicted by the Kalman filter, and flag as "suspicious" embedding steps where the difference would be too high. A combination of simulations and Planet-Lab experiments are used to demonstrate the validity, generality, and effectiveness of the proposed approach for two representative coordinate embedding systems, namely Vivaldi and NPS.

Finally, we addressed the question of guaranteeing the veracity of the coordinates advertised by nodes. Indeed, although if we secure the security of the embedding phase of Internet coordinate-based systems, or in other words the coordinate computation phase, these systems are above all intended to provide distance estimates between nodes, based on their coordinates only, even and all the more so if these nodes have never exchanged a distance measurement probe. Internet Applications would then benefit from an exchange of coordinates between nodes to estimate distances. However, these coordinate's exchanges would provide a malicious node with an opportunity to strike: in order to achieve some application-dependent goal or advantage (e.g. free-riding, denial-of-service, isolation, ubiquity gift, etc), a node can repeatedly lie about its coordinate. Simply lying about its coordinate could seriously disrupt the operations of such Internet applications relying on coordinate-based systems for distance estimation.

We then propose to leverage the Surveyor infrastructure and embedding cheat detection test to certify exchanged coordinates. More precisely, we propose that several Surveyors measure their distance to a node in order to verify the correctness of its claimed coordinate. If all Surveyors agree that this coordinate is the node's true coordinate, a time limited validity certificate, including the certified coordinate, is issued to the node.

Validity periods are computed based on an analysis of coordinate inter-shift times observed by Surveyors. Each surveyor estimates the time until the next shift and thus, and can then limit the validity of the certificate it issues to regular nodes for their calculated coordinates. Our method is illustrated on a trace collected from a Vivaldi system deployed on PlanetLab, where inter-shift times are shown to follow long-tail distribution (lognormal distribution in most cases, or Weibull distribution otherwise).

we then study the impact of a sophisticated attack on the accuracy of distance estimations, with and without our proposed coordinate certification defense mechanism. Carrying out such an attack, on a Vivaldi system deployed on PlanetLab, our method has been shown to be effective, enjoying good verification test performance (high true positive detection rate with low false positive rates), while achieving a very good trade-off between scalability and security. Indeed, the validity periods of certificates are rarely over-estimated, while they still do not trigger too frequent re-certifications. This work thus complements our detection protocol for embedding security.

1.3 Thesis Organization

The dissertation is structured as the following. In the next chapter, we present an overview of the proposed embedding coordinates systems in the literature. Particularly, we focus on those trying to equip their mechanism by a ”specific” detection mechanism against malicious nodes, and describe in more details the workings of the systems chosen for our study.

In Chapter 3, we first identify and classify the attacks, then we demonstrate and study the effects of theses attacks, through extensive simulations.

In Chapter 4, we present a general model of coordinate embedding, in the absence of malicious nodes, that naturally leads to the Kalman filter framework. We validate the model, with both simulations and PlanetLab experiments, in the case of both Vivaldi [15] and NPS [14]. This chapter also studies the viability of the idea of using Surveyor nodes in secure coordinate embedding.

In the last part of this chapter, we describe and evaluate, how Surveyors can effectively be used for malicious node detection in the specific embedding process of Vivaldi and NPS. Finally, the dissertation is concluded in chapter 6 with perspectives on our contributions and on future research in this area.

In Chapter 5, we address the issue of asserting the accuracy of Internet coordinates advertised by nodes of Internet coordinate systems during distance estimations. We study and characterize the coordinate inter-shift times and show that these times observed at Surveyors can adequately and statistically model inter-shift times at closeby nodes. Then we detail the certification procedure, and the performance evaluation of our proposal in the context of various attacks.

Finally, the conclusions of this thesis appear in chapter 6.

2

AN OVERVIEW OF NETWORK POSITIONING SYSTEMS

2.1 Introduction

As innovative ways are being developed to harvest the enormous potential of Internet infrastructure, a new class of large-scale globally-distributed network services and applications such as distributed overlay network multicast [20, 8], content addressable overlay networks [19, 21], and peer-to-peer file sharing such as Gnutella [22], OceanStore [7], BitTorrent [23, 24], etc. have emerged.

Because these systems have a lot of flexibility in choosing their communication paths, they can greatly benefit from intelligent path selection based on network performance. Collecting up-to-date latency measurements between nodes in an overlay network would be very beneficial for such applications. Especially, in a wide-area network, communication latencies have a significant impact on the overall execution time of operations.

For example, in a peer-to-peer file sharing application, a client ideally wants to know the available bandwidth between itself and all the peers that have the wanted file. Proximity-aware distributed hash tables would also use latency measurements to reduce the delay stretch of lookups [6]. Content distribution systems would construct network-aware trees to minimize dissemination times [25]. Decentralized web caches need latency information to map clients to cache locations, or to guide the selection of a download server from multiple replicas. And finally, a topology knowledge would allow the construction of efficient multicast delivery trees.

In this chapter, we present a briefing on different proposals for network positioning systems, with emphasis on the Internet coordinate-embedding systems, which we classify in two major classes: Landmark-based approaches, and decentralized Internet coordinates systems. The outline of this chapter is as follows. Next, we begin by describing several proposed works that do provide network proximity or location estimates, but do not rely on "virtual" coordinates embedded into geometric spaces. These systems are called direct measurements systems. Then, we concentrate on describing coordinate-based systems that fit within the landmark-based approaches for Internet positioning. Finally, we present different decentralized coordinate-based systems for network positioning. In particular, we focus on those systems including and using security mechanisms to try and filter out malicious nodes, although (as demonstrated in this dissertation) these are still rather primitive or still in their infancy and definitely cannot defend against all types of attacks.

2.2 Direct measurements Systems

Several approaches in the literature provide network proximity or location estimates using either direct pair-wise measurements, or by supplying applications with network distances estimates directly. In contrast to coordinates-embedding systems, these approaches do not attempt to globally model Internet hosts positions using absolute coordinates, but rather, most of them try to contribute in specific application needs, such as special peer lookups, or clustering, etc. In the following, we underline the mostly known approaches that have been proposed for locating network nodes.

2.2.1 Geolocation approaches

Many works have been proposed for inferring the geographical location of network nodes, rather than the Internet positions (e.g. in a latency space). We mention the following two interesting approaches:

- Constraint-Based Geolocation (CBG) approach [26] infers the geographical location of network nodes using multilateration. It consists in estimating the geographical position of a node using its geographical distances to a set of landmarks. Geographical distances to the landmarks are deduced from the correspondent delay distances (obtained by direct probing between the landmarks and the target host) by relying on the assumption that digital information travels along fiber optic cables at almost exactly 2/3 the speed of light in a vacuum. Basically, given the geographical locations of the landmarks and their geographical distances to a given target host, an estimation of the location of the target host is achieved using multilateration, as done with the Global Positioning System (GPS) [27].

- IP2Geo system [28] estimates the physical location of a remote server using information from the content of DNS names, whois queries, pings from fixed locations, and BGP information.

2.2.2 IDMaps

The first complete system that is aimed at predicting Internet distance is called IDMaps [29], which can be seen as the predecessor of landmark-based coordinate systems. IDMaps is an infrastructural service in which special HOPS servers maintain a virtual topology map of the Internet consisting of end hosts and special hosts called Tracers. This virtual topology map is used to predict Internet distance. For example, the distance between hosts x and y is estimated as the distance between x and its nearest Tracer T1, plus the distance between y and its nearest Tracer T2, plus the shortest path distance from T1 to T2 over the Tracer virtual topology. As the number of Tracers grow, the prediction accuracy of IDMaps tends to improve. Designed as a client-server architecture solution, end hosts can query HOPS servers to obtain network distance predictions.

Comparing to the IDMaps service, coordinate-based positioning systems are different in that nodes are enabled to use their own resources to compute their positions in the Internet. Moreover, these systems do not not directly interact with any applications. It is up to the applications running on end hosts to decide how to use the computed locations (coordinates).

2.2.3 Binning method

The Binning method aims to cluster nearby nodes using a technique of landmarks ordering. The binning method [30] consists then in clustering the nodes into bins where a bin is identified by a specific increasing order of closeness to landmark nodes. Thus, each ordering of landmarks represents a bin (e.g., for $N = 4$, $L_2L_4L_1L_3$ represents a bin). A node measures its RTT to each landmark and ranks the landmarks in an increasing order of RTT. Each node belongs therefore to the bin which has the same ordering of landmarks. It is expected that the nodes of the same bin are relatively closer to each other than to those in the other bins. Moreover, it is expected that the more the order of two bins is different the farther are their nodes.

The representation of a bin can be refined to use not only the ordering of landmarks but also the absolute value of the RTT measurements. These values can be indicated by dividing the range of possible delay values into a number of levels. For example, the range of possible delay values can be divided into the three following levels: (i) level 0 for delays in the range [0,100]ms, (ii) level 1 for delays in the range [100,200]ms, and (iii) level 2 for delays longer than 200ms. Thus, a bin can be identified by this notation: $L_iL_jL_k : a_ia_ja_k$ where $L_iL_jL_k$ is the relative ordering of landmarks and $a_ia_ja_k$ are the delay levels of each ordering respectively.

Application

One potential application that can profit from such clustering method is the topologically aware construction of overlays. Based on the binning strategy, authors in [30] construct the overlay network CAN [21] (Content-Addressable Network) to be congruent with the underlying IP topology. CAN is an application-level network forming a virtual d-dimensional cartesian coordinate space. This space is partitioned among all the overlay nodes in a manner that each node is responsible of a space portion. Routing in CAN consists in sending packets along the direct line path through the cartesian space from the source to the destination .

We briefly explain how the CAN overlay network can be constructed using the binning scheme. This scheme consists in dividing the space into the number of possible orderings of landmarks. Thus, there are $N!$ equal sized portions when the number of landmarks is equal to N. Each portion belongs to a single ordering. The partitioning of the space is done in a cyclical ordering of the dimensions (e.g., xyzxyzx...). First, the space is divided into N portions along the first dimension. Then, along the second dimension, each portion is divided into $(N - 1)$ portions each of which is divided to $(N - 2)$ portions and so on. A CAN node determines its bin based on its RTT measurements to the landmarks. Then, the node joins the CAN in a random point of the correspondent portion of the coordinate space which corresponds to its landmark ordering. When a node joins a portion which is already assigned to another node, the portion is divided equally between them. The problem is that this behavior may achieve an unequal dividing of the space among the overlay nodes. Even though the average number of network hops on the path between two nodes decreases when using such space partitioning, the load balancing between the nodes may be affected.

2.2.4 Hotz method

Hotz [31] estimates the distance between two nodes based on the delay vectors (i.e., delay measurement to a set of reference nodes) of these nodes. It consists in bounding the distance $D(p_i, p_j)$ (or the delay) between two nodes p_i and p_j by two values which are the difference distance $|D(p_i, L) - D(L, p_j)|$ and the summation distance $(D(p_i, L) + D(L, p_j))$ as shown in the following equation:

$$|D(p_i, L) - D(L, p_j)| < D(p_i, p_j) < (D(p_i, L) + D(L, p_j))$$

where $D(p_i, L)$ and $D(L, p_j)$ are the distances between a landmark L with nodes p_i and p_j respectively. Thus, if there are N landmark nodes then the delay between two nodes p_i and p_j is lower bounded by $max|D(p_i, L_k) - D(L_k, p_j)|$ and upper bounded by $min|D(p_i, L_k) + D(L_k, p_j)|$ where k varies from 1 to N as shown in the following equation:

$$max_{k=1...N}|D(p_i, L_k) - D(L_k, p_j)| < D(p_i, p_j) < min_{k=1...N}(D(p_i, L_k) + D(L_k, p_j)).$$

Thus, the distance between the nodes p_i and p_j can be estimated as the average value of these two bounds. One would need to perform such direct measurements to the landmarks at each distance inference between any pair of nodes in the Internet.

2.2.5 Traceroute-based approaches

Many approaches are designed and implemented to infer the network topology using the basic idea of the traceroute tool. In the following, we describe Rocketfuel, as one of the most popular traceroute-based approaches, incorporating different techniques from its predecessors.

Rocketfuel

Rocketfuel [32] infers the topology using traceroute probes, BGP routing tables, and DNS information. It aims mainly to find out the useful prefixes to be probed for the discovery of an ISP topology, then to collect the aliases which belong to the same router, using as few measurements as possible. A BGP table maps a destination IP address prefix to a set of ASes that are traversed to reach that destination. Thus, the use of BGP tables permits to pick up the prefixes whose traceroutes transit the ISP being mapped. Based on BGP tables, Rocketfuel defines three classes of traceroutes that transit the ISP network, then, it filters out the prefixes which are likely to follow redundant paths through the ISP network. It relies on the DNS information to check up if an obtained router address belongs to the ISP being mapped. This information serves to identify the router role and location, but requires costly measurements.

2.2.6 Meridian Approach

In [33], authors proposed a framework, called Meridian, for hosts to lookup their nearest peers in an overlay network. Each Meridian node keeps track of a small, fixed number of other hosts that are organized and maintained as a multi-resolution ring structure with exponentially growing ring radii. A node's query for its nearest peer can then be forwarded along the ring structure, which exponentially reduces the distance to the target at each query hop.

In contrast to coordinate-based systems, Meridian builds many local coordinates systems instead of a global coordinate system: This work focuses more on overlay construction and lookups, rather than distance prediction.

2.2.7 Application of locality in overlay-multicast

To evaluate how effective and scalable would be an overlay, using such approaches, we propose in [34, 35], a practical solution for large-scale and efficient multicast support, named LCC (Locate, Cluster and Conquer). First, we propose a simple and accurate location-aware process for connecting new members to the overlay network. The basic idea is to use a few nodes already involved in the constructed overlay to suggest candidate neighbors that are close to a newcomer. The latter gradually requests the suggested nodes to refine its localization in the underlying network. This locating process aims to be practical since nodes run the process locally in a 'stand-alone' fashion, and hence they do not rely on fixed landmarks measurements (needing dedicated deployed entities).

Second, we build a robust and scalable topology-aware clustered hierarchical overlay on the basis of the locating process. We propose proactive mechanisms to react to cluster leaders failures, and to smoothly manage overlay topology changes caused by crash scenarios or underlying network changes. Scalability is achieved by drastically reducing the bandwidth requirements for overlay maintenance. Robustness is obtained by mitigating the effect of dynamic environment as most changes are quickly recovered and not seen beyond clustered set of nodes. Running the locating process before that nodes join the overlay, and then clustering nearby nodes, allows to perform fast convergence to an efficient multicast delivery tree. Furthermore, it reduces management overhead and delivery tree changes imposed due to periodical refinements.

We carried out both simulations and PlanetLab experimentations to assess the effectiveness of the proposed techniques in LCC for large scale overlays, and to illustrate the system performance under particular real-world environments. Results show that our sheme incurs low overhead during both localization and data distribution. Compared to other enhancement-based and topology-aware approaches, our findings show that LCC achieves shorter convergence time and performs less link adjustments rate. At the same time, the scheme is robust to nodes' failures and performs well in terms of data distribution efficiency especially in large overlays.

However, this experience has shown its limits when it comes to exploit the locating process for multi-sessions applications. Running x applications that would rely on our locating process, would obviously result in x times measurement overhead. Moreover, this technique is specifically designed to look up for a node's closest neighbor, rather than locating the node itself according to all other nodes in the system. i.e. if one needs to determine the distance between a node and any other node in the system, we still need to perform on-demand direct measurements.

The following section discusses the major challenges of using such locating process, and more generally those of using direct measurements services.

2.2.8 Main drawbacks

Different approaches we discussed above try to solve either distance prediction or topology-aware routing problems with direct measurements. Meridian, for example, finds the nearest overlay node (i.e., one running Meridian) to an arbitrary node in the Internet through a large set of pings in direct response to an application-level request. In essence, this class of work solves the specific neighbor selection and overlay routing problems reactively, through a spike in activity in response to an application driven demand, while a long-running coordinate space would solve them pro-actively (as we will discuss it the following sections).

Although dynamic network performance characteristics such as available bandwidth and latency are the most relevant to applications and can be accurately measured on demand, the huge number of wide-area-spanning end-to-end paths that need to be considered in these distributed systems makes performing on-demand network measurements impractical because it is too costly and time-consuming.

To exploit network locality, if overlay networks were left with the burden of performing their own network measurements, implementations that gather all-pairs latency measurements would only be scalable for relatively small overlay deployments. For example, the all-pairs ping service managed by Stribling[39] has recently ceased operation because it became infeasible to obtain up-to-date measurements for over 500 PlanetLab nodes.

Proximity measurements, based on repeated pair-wise distance measurements between nodes, can prove to be very onerous in terms of measurement overheads. Indeed, the existence of several overlays simultaneously can result in significant bandwidth consumption by proximity measurements (i.e. ping storms) carried out by individual overlay nodes [40]. Also, measuring and tracking proximity within a rapidly changing group requires high frequency measurements. In contrast, the embedding techniques do not require a full mesh of RTT measurements, to predict distance between any pair of nodes in the system.

In other words, most of the systems we introduced above are dedicated to overlay construction and lookups, rather than distance prediction at the Internet scale in a timely fashion. In order to predict distances between any pair of nodes in the Internet, these non coordinate-based systems still need to perform costly measurements.

2.2.9 Why using Internet Coordinate-based positioning systems?

To bridge the gap between contradicting goals of performance optimization and scalability, several coordinate-based approaches to predicting network distances (as round trip propagation and transmission delay) have been proposed. As illustrated in

Figure 2.1, the key idea of such systems is to model the Internet as a geometric space

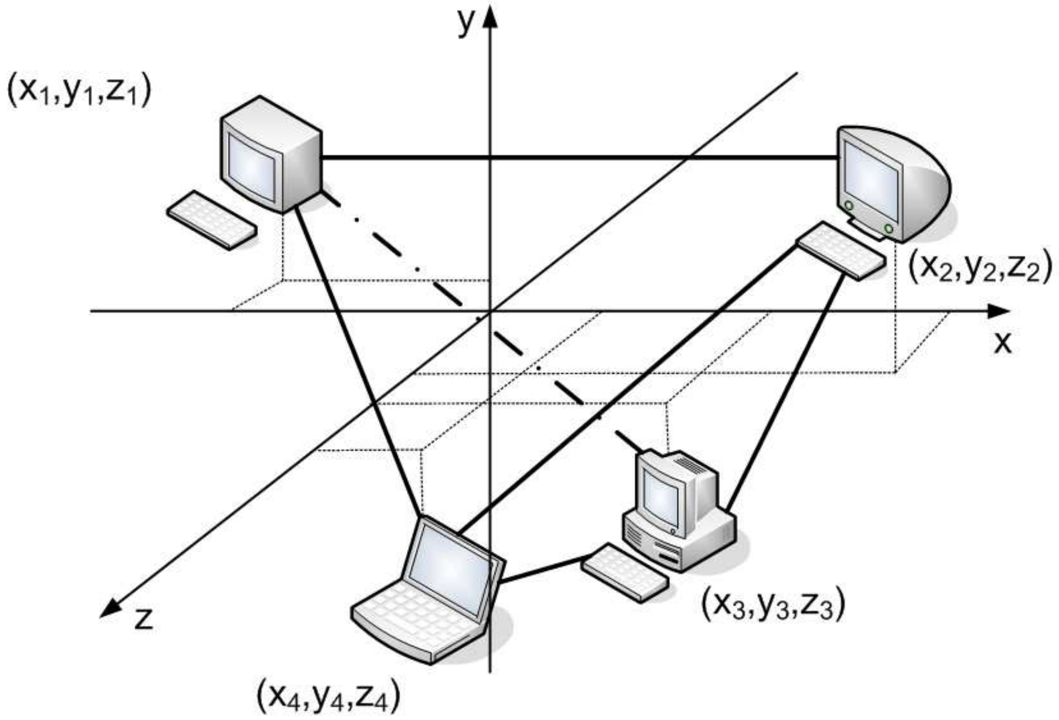


Figure 2.1: Geometric space model of the Internet

(e.g. a 3-dimensional Euclidean space) and characterize the *position* of any node in the Internet by a position in this space. The network distance between any two nodes is then predicted as the geometric distance between their coordinates without explicit measurements. In other words, if a node x learns the coordinate of a node y, x does not have to perform an explicit measurement to determine the RTT to y; instead, the distance between x and y in the coordinate space is an accurate predictor of the RTT. In other words, as long as a reasonably accurate position for a node can be obtained with little effort, much of the distance measurement sampling cost can be eliminated and the remaining overhead amortized over many distance predictions.

Ultimately, coordinate-based systems for network positioning aim to provide a network positioning capability to all hosts in the Internet in a scalable and a timely fashion.

The Key distinguishing properties of such predicting network distances approaches are then the following:

- **Peer-to-peer architecture friendly**: Coordinate-based systems can naturally be incorporated into peer-to-peer applications since end hosts (nodes) can easily maintain geometric coordinates that characterize their locations in the Internet.

- **Extremely fast**: When an end host discovers the identities of other end hosts in a peer-to-peer application, their pre-computed coordinates can be piggybacked, thus network distances can essentially be computed instantaneously by the end host. There is no additional communication delay whatsoever. The off-line computation of node coordinates is also very simple and fast.

- **Highly scalable**: Another benefit of coordinate-based systems is that coordinates are highly efficient in summarizing a large amount of distance information. For example, in a multi-party application, the distances of all paths between K nodes can be efficiently communicated by K sets of coordinates of D numbers each (i.e. $O(KD)$ of data), as opposed to $K(K-1)/2$ individual distances (i.e., $O(K^2)$ of data). Thus, this approach is able to trade local computations for significantly reduced communication overhead, achieving higher scalability as demonstrated in [11].

- **Structured representation**: The geometric coordinates of nodes describe a simple and yet highly structured representation of the complex Internet topology. Many algorithms can then take advantage of this structure to perform topologically aware operations on the Internet in a scalable fashion.

2.3 Fixed Landmark-based Coordinate Systems

These systems involve a centralized component (a set of landmark nodes), where other nodes compute their coordinates according to measurements to this already deployed fixed infrastructure. The concept of landmark-based positioning was initially introduced in [11] for the design of the Global Network Positioning (GNP) system.

2.3.1 GNP: Global Network Positioning

In such a system, the coordinates of the landmarks are first computed by minimizing the error between the measured distances and the estimated distances among the landmark nodes. An ordinary node derives its coordinate by minimizing the error between the measured distances and the estimated distances to the landmarks. More formally, the method works as follows. N special nodes called Landmarks, denoted by $LM_1, ..., LM_N$ are deployed in the Internet and the inter-Landmark distances are measured. The inter-Landmark distances are transmitted to a central node. The central node then computes a set of Landmark coordinates and return the coordinates to the Landmarks. The Landmark coordinates, denoted by $c_{L_1}, ...c_{L_N}$, are the result of minimizing the following objective function:

$$f_{obj1}(c_{LM_1}, ...c_{LM_N}) = \sum_{i,j \in 1,...,N | i>j} \epsilon(d_{LM_i LM_j}, \hat{d}_{LM_i LM_j})$$

where $d_{LM_i LM_j}$ is the *measured distance* between LM_i and LM_j, $\hat{d}_{LM_i LM_j}$ is the geometric distance between c_{LM_i} and c_{LM_j}, that we will call *virtual distance* [1], and $\epsilon(.)$ is the error measurement function:

$$\epsilon(d_{H1H2}, \hat{d}_{H1H2}) = (\frac{d_{H1H2} - \hat{d}_{H1H2}}{d_{H1H2}})^2$$

GNP uses the Simplex Downhill algorithm [41] to solve the optimization problem. The Landmarks' coordinates define the bases for the geometric space. To embed an ordinary node H, H uses the Landmarks as reference points and probes all the Landmarks to obtain the Landmarks' coordinates and the network distances to the Landmarks (see Figure 2.2). It then computes its coordinate, c_H, that minimize the following objective function:

$$f_{obj2}(c_H) = \sum_{i \in 1,...,N} \epsilon(d_{HLM_i}, \hat{d}_{HLM_i})$$

Using this method with Internet measurements, it has been shown in [11] that the Internet network distance relationship is accurately "embeddable" in a low dimensional Euclidean space. In particular, in a 7-dimensional Euclidean space, 50% of the distance predictions have less than 10% error, 90% of them have less than 50% error, and the network positions can be used to accurately select nearby nodes in the Internet.

Next, we present different extensions of the concept of Landmarks introduced by GNP, that have been proposed in the literature.

[1] denoted also in the rest of this thesis, the predicted distance in the geometric embedding space

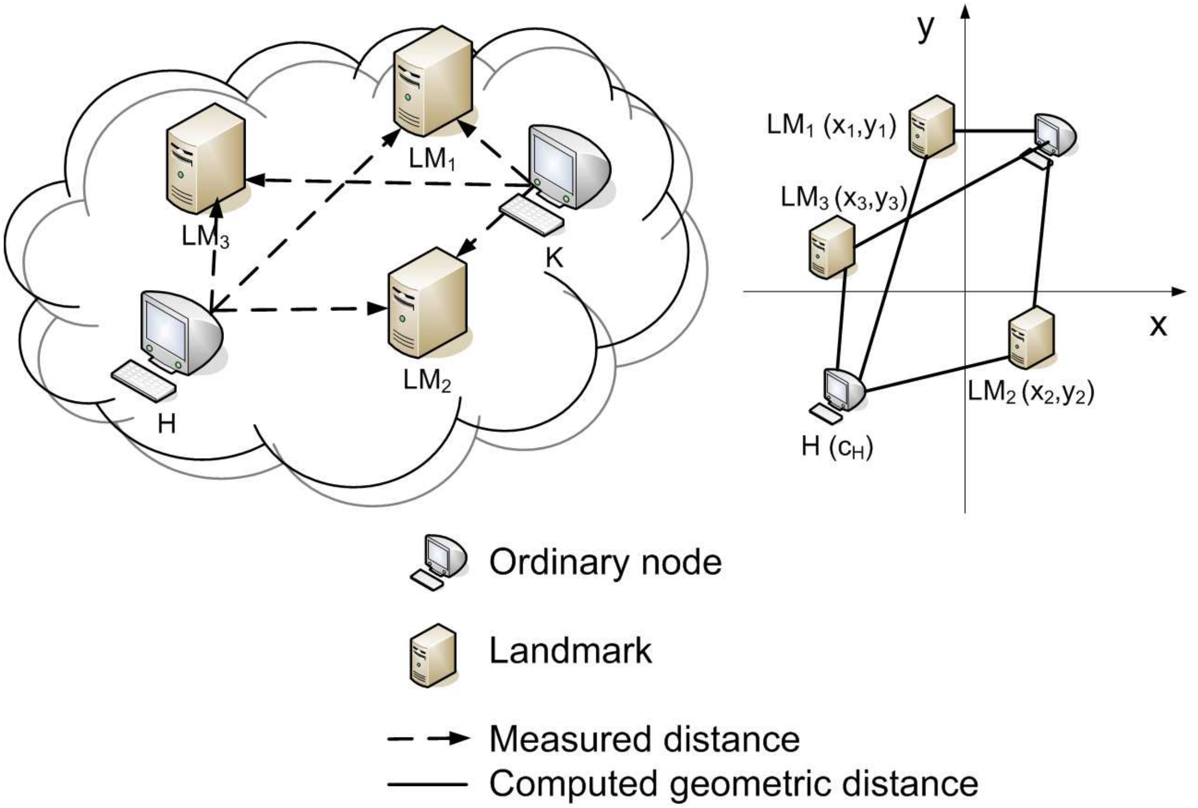


Figure 2.2: Basic network positioning method in a 2-d Euclidean space

2.3.2 Lighthouse

Lighthouse [12] is an extension of GNP that is intended to be more scalable. Although it has a special set of landmark nodes, a joining Lighthouse node does not have to query those global landmarks. Instead, it can query any existing set of nodes to find its coordinate relative to that set, and then transforms those coordinates into coordinates relative to the global landmarks. This flexibility is allowed provided each node maintains a transition matrix for global basis G. It uses linear matrix factorization such as the QR decomposition (Gram-Schmidt orthogonalization).

2.3.3 Virtual Landmarks

Tang and Crovella propose a coordinate system based on “virtual” landmarks.

In [42], the authors propose the use of the Lipschitz embedding [43] in order to embed distances between nodes in a low dimensional space obtained by compressing the full delay matrix using the PCA method (Principle Component Analysis [44, 45]).

The basic idea of the Lipschitz embedding is to use network distances themselves as coordinates. To find the coordinate vector $\overrightarrow{x}_i$, for node i, one sets the j^{th} component of $\overrightarrow{x}_i$ to the measured distance between node i and landmark j, for $j = 1, ..., n$.

The Lipschitz embedding can be accurate because two entities that are close to each other in a metric space typically have similar distances to many other entities. Thus two nearby points in the original metric space may have very similar coordinate vectors, and so may map to nearby points under the Lipschitz embedding.

This study also explores methods to reduce the number m of Landmarks that need to be probed without adversely affecting the accuracy.

By applying the PCA method [44, 45] to an $m \times n$ matrix A in which row i is the initial n-dimensional coordinate vector $\vec{x}_i$ for node i, we can map each $\vec{x}_i$ to a new $\vec{y}_i$ in a lower dimensional space, while approximately preserving distances. That is we map $\vec{x}_i \in \mathbb{R}^n$ to $\vec{y}_i \in \mathbb{R}^r$ such that $r \ll n$ and $||\vec{x}_i - \vec{x}_j|| \approx ||\vec{y}_i - \vec{y}_i||$ for all i, $j \in$ 1, ..,m.

The mapping from $\vec{x}_i$ to $\vec{y}_i$ obtained via PCA is a linear one. That is, $\vec{y}_i = M\dot{\vec{x}}_i$ for some M (where M is an $r \times n$ matrix). Final coordinate of node i (the components of $\vec{y}_i$) can be seen as distances to virtual landmarks. The distance to a virtual landmark is defined as a linear combination of distances to actual landmarks.

A parallel study [46] has also suggested the use of the PCA method to compute network positions. This study analyzes the difficulty in the nonlinear optimization of Euclidean embedding, and shows that a PCA-based schemes is more computationally efficient. Distributed versions of these methods however remain to be studied and developed.

2.3.4 NPS

The Network Positioning System (NPS) [14] extends GNP into a hierarchical coordinate system, where all nodes could serve as landmarks (reference points) for other nodes. It aims to recover "gracefully" from either landmark failures, or situations where these special entities of the system and their network access links become performance bottlenecks. The main departure from GNP is that any node that has determined its position can be chosen by a membership server to be a reference point for other nodes. Actually, the membership server randomly chooses eligible nodes to become reference points when the permanent landmarks are too heavily loaded or unavailable. However, to ensure consistency, NPS imposes a hierarchical position dependency among the nodes (see Figure 2.3).

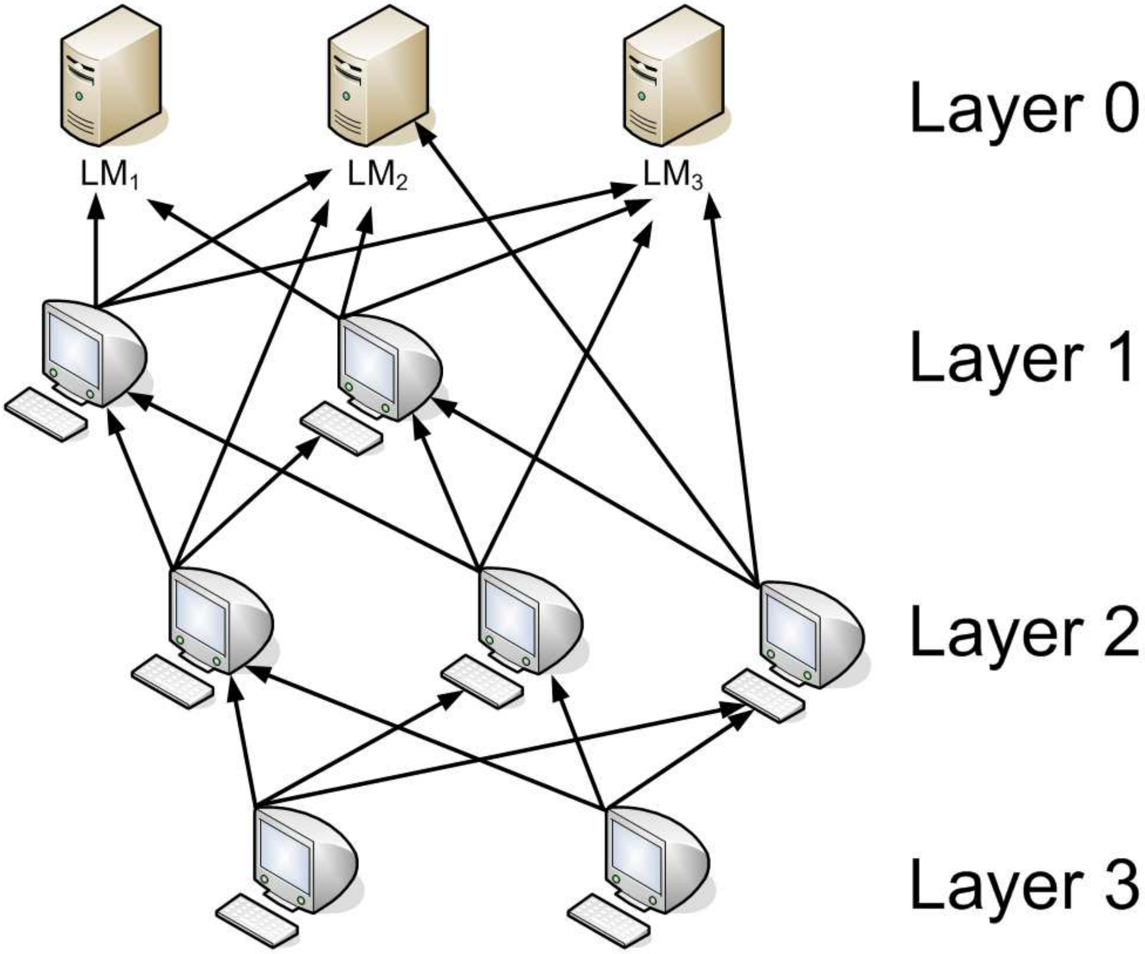


Figure 2.3: The hierarchical network positioning architecture of NPS.

Given a set of nodes, NPS partitions them into different layers. A set of 20 landmarks are placed in layer-0 (or L_0), the top layer of the hierarchy (these permanent landmarks are the fixed infrastructure used to define the bases of the Euclidean space model), and an 8-dimension Euclidean space is used for embedding. Each node in layer L_i randomly picks some nodes in layer L_{i-1} as its reference points. The relative error of the distance prediction between a pair of nodes is defined as:

$$relative\ error = \frac{|actual - virtual|}{min(actual, virtual)}$$

In [14], authors argue that a 3-layer NPS system is already very accurate and can support more than 2 billion nodes.

NPS includes a strategy for mitigating the effects of simple malicious attacks. Indeed, malicious nodes could potentially lie about their positions and/or inflate network distances by holding onto probe packets. The basic idea is to eliminate a reference point if it fits poorly in the Euclidean space compared to the other reference points. Each node, when computing its coordinate, based on measurements from different reference points, would reject the reference that provides a relative error significantly larger than the median error of all other reference nodes. Specifically, assume there are N reference points R_i, at positions P_{Ri}, and the network distances from a node H to these are D_{Ri}. After H computes a position P_H based on these reference points, for each R_i, it computes the fitting error E_{Ri} as $\frac{|distance(P_H, P_{Ri}) - D_{Ri}|}{D_{Ri}}$. Then the requesting node, H, decides whether to eliminate the reference point with the largest E_{Ri}. The criterion used by NPS is that if:

$$\max_i E_{Ri} > 0.01 \tag{2.1}$$

and,

$$\max_i E_{Ri} > C \times median_i(E_{Ri}), \tag{2.2}$$

where C is a sensitivity constant, then the reference point with $max_i E_{Ri}$ is filtered (i.e. H tries to replace it by another reference point for future repositioning).

In this thesis, we studied NPS as a representative of the landmark-based coordinate systems. More specifically, in the next chapter, we will show how this infrastructure-based system can, under some well chosen attack strategies, be vulnerable to sophisticated attacks that are aimed at defeating its defense mechanism.

2.4 Decentralized Internet Coordinate Systems

This class of approaches extends the embedding concept, either by generalizing the role of landmarks to any node existing in the system, or by eliminating the landmark infrastructure. Decentralized Internet coordinate systems can be seen as peer-to-peer network positioning systems.

2.4.1 Practical Internet Coordinates: PIC

Practical Internet Coordinates (PIC) [13] is one of the recent decentralized coordinate systems using again the Simplex Downhill method to minimize an objective distance error function (sum of relative errors). It does not require explicitly designated landmarks. In PIC, the joining node can pick any node whose coordinate have already been computed to be a landmark. It uses an active node discovery protocol to find a set of nearby nodes to use to compute coordinates. Different strategies such as random nodes, closest nodes, and a hybrid of both, are proposed.

PIC aims to defend the security of its coordinate system against independent malicious participants using a test based on the triangle inequality. Basically, the test relies on the observation that the triangle inequality holds for most triples of nodes in the Internet. Therefore, PIC assumes that for most triples of nodes a, b and c , $d(a,b)+d(b,c) \geq d(a,c)$, where $d(i,j)$ denotes either the measured network distance between nodes i and j or the virtual distance in the geometric space.

The intuition behind the security test of PIC is as follows. An attacker that lies about its coordinate or its distance to the joining node is likely to violate triangle inequality. The joining node uses the distances it measured to each landmark node and the coordinates of the landmarks to check for violations of the triangle inequality. It then removes from its proper set of landmarks used for positioning, the nodes that most violate the triangle inequality.

Let's denote by n a new node joining PIC, that is computing its coordinate using i and j, two distinct Landmarks. This is illustrated in figure 2.4. If $d(n,i)$ is the measured distance between n and i and $d'(i,j)$ is the predicted virtual distance (using coordinates) between i and j, all of the following inequations should hold:

$$d(n,i) \leq d(n,j) + d'(i,j)$$

$$d(n,i) \geq d(n,j) - d'(i,j)$$

$$d(n,i) \geq d'(i,j) - d(n,j)$$

The first inequality imposes an upper bound on the measured distance to i and the other two impose a lower bound of $|d(n,j) - d'(i,j)|$. For each landmark used for coordinates computation, the security test checks whether the upper bounds and lower bounds defined by each landmark j are satisfied by i and computes the following two metrics:

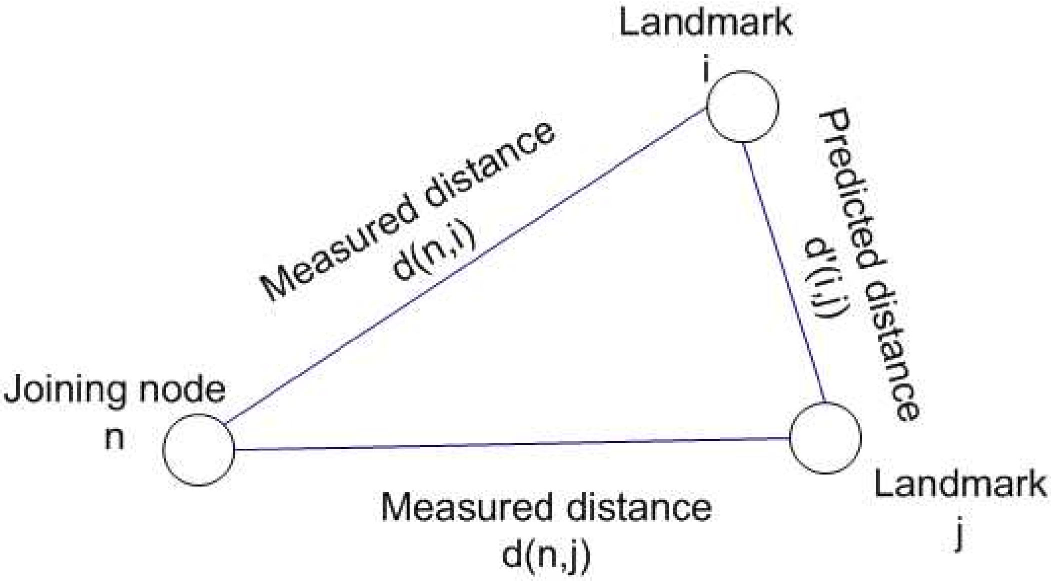


Figure 2.4: Triangle inequality with measured and predicted distances in PIC.

$$upper_i = \sum_{j=1}^{N} \begin{cases} d(n,i) - (d(n,j) + d'(i,j)) & \text{if } d(n,j) + d'(i,j) < d(n,i), \\ 0 & \text{Otherwise} \end{cases}$$

$$lower_i = \sum_{j=1}^{N} \begin{cases} |d(n,j) - d'(i,j)| - d(n,i) & \text{if } |d(n,j) - d'(i,j)| > d(n,i), \\ 0 & \text{Otherwise} \end{cases}$$

where N is the number of distinct Landmarks used by n for positioning in the system. $upper_i$ is the sum of the deviations above the upper bounds, and $lower_i$ is the sum of the deviations below the lower bounds. The security test computes the maximum value of both metrics for all Landmarks used by n and removes the corresponding node. Then, the joining node uses the Simplex to compute its coordinates with the remaining landmarks. This process is repeated a fixed number of times.

In [13], authors show that such security test can deal with up to 20% of malicious nodes existing in the system. However, [49] and [48] indicate that network RTTs commonly and persistently violate the triangle inequality. A security mechanism based on the fact that the triangle inequality systematically holds, may degrade the performance of a clean system, without malicious nodes inside.

2.4.2 Vivaldi

Vivaldi [15] is a fully distributed algorithm, requiring no fixed network infrastructure and no distinguished nodes. It is based on a simulation of springs, where the position of the nodes that minimizes the potential energy of the springs also minimizes the embedding error.

A new node computes its coordinates after collecting latency information from only a few other nodes. Basically, Vivaldi places a spring between pairs of nodes (i, j) with a rest length set to the known (measured) RTT between them. The current length of the spring is considered to be the distance between the nodes as estimated in the coordinate space. The potential energy of such a spring is proportional to the square of the displacement from its rest length: the sum of these energies over all springs is the error function that Vivaldi nodes try to minimize.

An identical Vivaldi procedure runs on every node. Each sample provides information that allows a node to update its coordinate. Each sample used by a node i, is based on measurement to a remote node j, its coordinate x_j and the estimated error reported by j, e_j. The algorithm handles high error nodes by computing weights for each received sample.

The relative error of this sample, e_s, is then computed as follows:

$$e_s = | \parallel x_j - x_i \parallel - RTT_{measured} | \; / \; RTT_{measured}$$

The node then computes the sample weight balancing local and remote error : $w = e_i/(e_i + e_j)$, where e_i is the node's current (local) error. This sample weight is used to compute an adaptive timestep, δ defining the fraction of the way the node is allowed to move toward the perfect position for the current sample: $\delta = C_c \times w$, where C_c is a constant fraction < 1. The node then updates its local coordinate as follows:

$$x_i = x_i + \delta \cdot (RTT_{measured} - \parallel x_i - x_j \parallel) \cdot u(x_i - x_j)$$

where $u(x_i - x_j)$ is a unit vector giving the direction of i's displacement. Finally, it updates its local error as $e_i = e_s \times w + e_i \times (1 - w)$.

Vivaldi considers a few possible coordinate spaces that might better capture the underlying structure of the Internet. Coordinates embedding maps the network distances into different geometric spaces, for instance 2D, 3D or 5D Euclidean spaces, spherical coordinates, etc. Vivaldi also introduces the *height model*, consisting of a Euclidean coordinate space augmented with a height. The Euclidean portion models a high-speed

Internet core where latencies are proportional to geographic distance, and the height models the time it takes packets to travel the access link from the node to the core.

In [15], authors show that the more dimensions a Euclidian space has, the more accurate the Vivaldi system is. Moreover, results show that height vectors perform better than both 2D and 3D Euclidean coordinates, as the height model is a better approximation of the hyperbolic curvature of the Internet [47].

The reader should finally note that, although Vivaldi defends against high-error nodes, it does not address issues related to malicious behaviors. In the next chapter, we will study how such vulnerability can be exploited to spoil and seriously disrupt the operations of the Vivaldi system. The latter is studied in this thesis as a representative of decentralized Internet coordinate-based systems.

2.4.3 The Big-Bang Simulation: BBS

Big-Bang Simulation (BBS) [16] performs a simulation, similar to Vivaldi's mass-spring system, to calculate coordinates, simulating an explosion of particles under a force field. The simulation models each particle's momentum explicitly and then introduces friction in order to cause the simulation to converge to a stable state.

2.5 Discussion and Conclusions

Although coordinate-based systems have attractive properties for latency prediction on the Internet, one could criticize them for requiring expensive maintenance and having prediction accuracy worse than direct measurement methods such as Meridian. At the very least, triangle inequality violations [48] could be a major barrier for the accuracy of such systems. In [49], authors also observed that absolute relative error may not be the major indicator of the quality of an embedding as experienced by a user. They showed that, using other accuracy metrics that attempt to quantify various aspects of user-oriented quality (such as Relative Rank Loss or Closest Neighbors Loss), the quality of the coordinate-based systems is not as high as that suggested by the use of absolute relative error.

Moreover, choosing the suitable geometric space for coordinate embedding, and more generally, to model the Internet has received much attention by the research community, and has been shown to be a challenging task. Basically, coordinate systems have concentrated on pure Euclidean spaces or other simple geometric spaces like the surfaces of spheres and tori. In [47], authors introduced a new coordinate space that places nodes some distance “above” a Euclidean space (height model). Shavitt and Tankel [47] proposed using a hyperbolic coordinate space to model the Internet. The hyperbolic model may address a shortcoming of the Vivaldi’s height model that implicitly assumes that each node is behind its own access link. If two nodes are behind the same high-latency access link, the height model will incorrectly predict a large latency between the two nodes: the distance down to the plane and back up.

Nevertheless, current live implementations and deployments of Internet coordinates systems in the ’wild’, show that using such systems is beneficial for P2P applications, and overlays ([10, 51, 50]) that rely on the notion of network topology-awareness. In [18], using the Azureus BitTorrent network as a testbed, authors show that even if, live, large-scale network coordinate systems behave somewhat differently than the PlanetLab and simulation-based counterparts, Azureus’ coordinates and, by inference, Internet-scale coordinate-based systems in general, were able to tackle a basic goal: quickly and efficiently optimizing anycast decisions based on correct latency estimates [2]. Ledlie et al. showed that incorporating Vivaldi’s coordinates in a one million node Azureus network, improves the efficiency of the latter. However, this is achievable by implementing specific techniques in Azureus in order to support coordinates in an effective way. Basically, to improve the accuracy and stability of coordinate-based systems, several works proposed different techniques:

- latency filters and application-specific coordinate updates, in order to make the distinction between constantly evolving “system-level” coordinates and “useful application-level” coordinates that should be stable [18]. It should also be noticed that De Launois et al. [52] propose a different method for stabilizing coordinates: asymptotically dampening the effect of each new Vivaldi measurement. While this factor does mitigate oscillations in a fixed network, it prevents the algorithm from adapting to changing network conditions.

- gossip-based coordinates update, rather than piggybacked coordinates on to application-level messages. This technique has been shown to expand the size of the working set of Vivaldi, expanding the set of neighbors for each node, and then improving its accuracy [18].

- violator exclusion, inspired by the removal of a small percentage of the nodes

[2] Azureus [10] is currently one of the most popular clients for BitTorrent, a file sharing protocol [23]

with the largest triangle inequality violations from the Azureus latency matrix. Removing 0.5% percent of nodes leads to 20 percent improvement in global accuracy [18]. These observations confirm a theoretical work that showed how to decrease embedding distortion by sacrificing a small fraction of distances to be arbitrarily distorted [53]. These results mainly show that if a mechanism could prevent a small percentage of nodes (Triangle inequality violators) from affecting the rest of the system, it would improve overall accuracy.

Finally, it should be noted that if latency is the primary network metric that has been embedded in coordinate spaces, there are at least two approaches to including other network characteristics, such as bandwidth and jitter. These could be made as additional dimensions in existing latency space. For instance, it has been demonstrated that per node characteristics, such as load, can be included to form a cost space that allows hot-spot detection [54]. Second, Oppenheimer et al. and Lee et al. have investigated the inverse correlation between latency and bandwidth [55, 56]. The correlation Oppenheimer found implies that network-aware decisions made in the latency space may result in good bandwidth characteristics.

In conclusion, while developing coordinate-based systems with perfect accuracy is a long-term challenge, current approaches are already sufficiently accurate for most applications and allow trade-offs between accuracy and measurement overhead for dynamic topology-aware overlays.

But it should also be noticed that these come at the expense of slow convergence times ranging from tens of seconds to several minutes [18]. This is several orders of magnitude slower than what is achievable with direct ’on-demand’ distance measurements between nodes and is often unacceptable for topology-aware applications whose aim is to quickly identify “best nodes”.

We therefore contend that coordinate-based positioning systems are an attractive proposition if they are deployed as a service: every node could run a coordinate system daemon at boot time which would then be capable of providing accurate coordinate estimates to applications and their overlays on request. In essence, the coordinate system could then be seen as a component of a “virtual infrastructure” that supports a wide range of overlays and applications.

But a system providing an “always-on and large scale coordinate service” would also likely be a prime target for attacks, as its disruption could result in the mis-functioning or the collapse of very many applications and overlays. Indeed, as the use of overlays and applications relying on coordinates increases, one could imagine the release of worms and other malware whose purpose is to attack the virtual infrastructure as a whole.

Securing the base of distance prediction for many applications would then be much more critical, than detailing security of the artifacts of any particular application. Moreover, regardless of the accuracy of these Internet coordinate systems, securing them is a necessary condition to their deployment. In this context, this thesis provides the missing piece in the design of internet coordinates systems.

3

Disrupting Internet Coordinates Systems

3.1 Introduction

Most, if not all, of current proposals for coordinates systems assume that the nodes partaking in the system cooperate fully and honestly with each other, that is that the information reported by probed nodes is correct. This could also make them vulnerable to malicious attacks. In particular, insider attacks executed by (potentially colluding) legitimate users or nodes infiltrating the system could prove very effective. In this chapter, we will identify potential attacks against such coordinate-based systems, and we will focus on studying how attackers can destabilize both the Vivaldi and NPS coordinate systems: Vivaldi as a prominent representative of purely peer-to-peer-based (i.e. without infrastructure support) positioning systems, and NPS as typical of landmark-based systems.

3.2 Threats and Attack Classification

We classify attacks and identify threats that malicious nodes may seek to carry out on coordinate-based positioning systems. We consider malicious nodes that have access to the same data as legitimate users, often called *Insiders*. This means that participants are not completely trusted entities, or that malicious nodes have the ability to bypass any authentication mechanisms. Malicious nodes are able to send misleading information when probed, or send manipulated information after receiving a request or affect some metrics observed by chosen targets. The main classes of attacks on positioning system behavior are:

1. Disorder: the main goal of this attack is to create chaos as a form of denial of service (DoS) attack. This results in high errors in the positioning of nodes, or the non-convergence of the embedding algorithm. The attack would then just consist in maximizing the relative error of nodes in the system, either passively by not cooperating or falsifying its coordinates or by actively delaying probes.

2. Isolation: where nodes would be isolated in the coordinate space. The attack could target a particular node, in order to convince the victim that it is positioned in an remote zone of the network. The ultimate goal of such an attack can be, for instance, obliging the victim to connect to an accomplice node as the closest node in that zone, in order to perform traffic analysis or packet dropping, man in the middle attacks, etc. One way a malicious node can conduct this attack is to delay probes sent by the victim, and to falsify its own coordinates, so that the victim's computed coordinates are set to a value large enough, to be far from other nodes.

3. Repulsion: where a malicious node would convince its victims that it is positioned far from them in order to reduce its attractiveness, and then, for instance, alleviate its resource consumption by not cooperating in the application progress. Ways to perform such attacks are to make its conditions (performance, position) seem worse than they actually are. This is accomplished by means of delaying measurement probes and/or by manipulating the coordinates transmitted to other nodes.

4. System Control: This attack is possible on coordinate-based systems that allow "normal" nodes to be considered as landmarks, i.e. most of the existing systems except the centralized systems. In hierarchical systems for example, such as NPS, nodes would try to get higher in the hierarchy in order to fool and influence the maximum number of correct nodes.

These classes of attacks can either be carried out by malicious nodes in an independent manner or as a conspiracy created by colluding nodes. Collusion is likely in scenarios where attack propagation happens through the now well tested means used in today's DDoS attacks (e.g. worms, etc.).

It should be noted that all attacks, be they explicitly aimed at disrupting the whole system or skew the coordinates of a single node, will often result in some distortion of the coordinate space. This is because of the possible cooperation between the nodes that will act as a catalyzer to the propagation of errors to other (non directly targeted) nodes.

3.3 Performance Evaluation

3.3.1 Performance Indicators

We use the relative error (as defined specifically for each algorithm in the last chapter) as our main performance indicator. We compute the average relative error over all nodes to represent the accuracy of the overall system. Since our focus is on measuring the impact of malicious nodes on the system, we also introduce the *relative error ratio* (also called "Ratio" for simplicity), which is the relative error measured in presence of malicious nodes normalized to the performance of the system without cheats used as the best case scenario (i.e. $\text{error\_ratio} = \text{error}/\text{error}_{\text{ref}}$). Obviously, a value for the error ratio above 1 indicates a degradation in accuracy.

As the worst case scenario, we also compute the relative error of a coordinate system where nodes choose their coordinates at random. In this random scenario, all nodes choose their coordinate components randomly in the interval $[-50000, 50000]$ (for each dimension of the coordinate).

In this chapter, we will concentrate on two systems: NPS (chapter 2, section 2.3.4) as a representative of the landmark-based approach; and Vivaldi (chapter 2, section 2.4.2) as a representative of the infrastructure-less approach.

We quantified the impact of the attacks we described above through simulations of different potential scenarios. In the following, we present the details of our simulation set-up.

3.3.2 Simulation Set-up

We used the "King" dataset to model Internet latencies based on real world measurements. This dataset contains the pair-wise RTTs between 1740 Internet DNS servers collected using the King method [57]. This was used to generate a topology with 1740 overlay nodes, from which we derived various group sizes by picking nodes at random (unless otherwise stated, in the simulations, the group consists of all the 1740 nodes). Each scenario was repeated 10 times with the malicious nodes selected at random within the group. We consider groups with 10%, 20%, 30%, 40%, 50% and 75% of malicious nodes. In view of the infection rates of recent worm epidemics, we believe these values to be realistic, both during and for a long time after an outbreak.

For the Vivaldi simulation scenarios, we used the p2psim discrete-event simulator [58]. Each Vivaldi node has 64 neighbors (i.e. is attached to 64 springs), 32 of which being chosen to be closer than 50 ms. The constant fraction C_c for the adaptive timestep (see section 2.4.2) is set to 0.25. These values are those recommended in [15]. The system is considered to have stabilized when all relative errors converge to a value varying by at most 0.02 for 10 simulation ticks. We observed that Vivaldi without malicious nodes always converged within 1800 simulation ticks, which represents a convergence time of over 8 hours (1 tick is roughly 17 seconds). Unless otherwise stated, our results are obtained for a 2-dimensional coordinate space.

For NPS, we developed our own event-driven network simulator, based on the description of the protocol in [14] and a reference implementation of the protocol[1]. Unless otherwise stated, as recommended in [14], we considered an 8-dimensional Euclidean space for the embedding. In layer-0, a set of 20 well separated permanent Landmarks are chosen. 20% of nodes are randomly chosen as reference points, in each subsequent layer. For the security mechanism of NPS, the sensitivity constant C was set to 4.

Finally, in this thesis, we consider all the attacks in an “injection” context, where the malicious nodes are introduced in a system that has already converged. This is in contrast with a “genesis” attack where the malicious nodes are present from the system’s creation time (which we studied in [1] for Vivaldi). The former is more realistic in a practical setting, generalizes the attack strategies’ impact, and reflects the emergence of threats carried out by malware in the Internet.

[1] I would like to thank Prof. Eugene Ng for sharing his code.

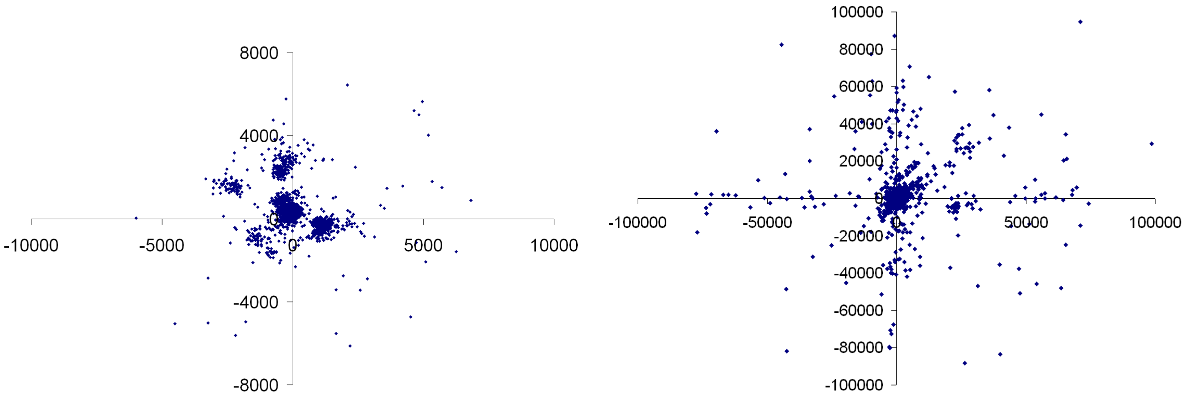
3.4 Attacks on Vivaldi

We first discuss ways to achieve disorder attacks in Vivaldi. As it is a fully-distributed algorithm relying on cooperation of nodes in order to ensure accuracy of the computed coordinates, it seems easy to fool honest nodes.

3.4.1 Disorder Attack

The disorder attack has no specific objective, but false coordinate computations and high positioning error. When solicited, a malicious node sends a randomly selected coordinate x_j, associated with a very low error, $e_j = 0.01$. Moreover, each node's measurement is delayed by a randomly generated value in [100..1000] ms. In this first scenario, it is not necessary to care about lie consistency, as Vivaldi uses error weights sent along with the responses to probes to adjust the adaptive timestep. Even if the measured distance $RTT_{measured}$ to malicious node j is not consistent with the coordinates x_j, the victim i would consider itself as a high error node, and would try to adjust its coordinates by a great adaptive timestep value, due to the fact that j sends a low error.

Figure 3.1 illustrates the effects of malicious nodes on the coordinate space of Vivaldi. we can see that the topology we used exhibits a clear "cluster" structure that disappear in the presence of only 10% of malicious nodes. This is because, in this disorder attack, the attackers keep "jolting" the system and the errors introduced "ripple"



(a) Vivaldi

(b) Vivaldi with 10% malicious

Figure 3.1: *Effect of Disorder attack on Vivaldi with 1740 nodes* through the system, propagated through normal operations of the honest nodes.

Figure 3.2 depicts the relative error ratio variation in function of time, for our full set of 1740 nodes, representative of the impact of the malicious nodes on the system. It is clear that enough attackers can quickly destabilize a converged system and seriously reduce the system accuracy. It is interesting to note that, in the presence of enough malicious nodes, despite the system converging in the sense that the relative errors at each node stabilize, these errors are so high that a great variation of the coordinates of a node barely affects the associated error. In other words, the coordinates of the nodes keep showing great variations and do not stabilize but the error introduced by such constant movement is stable because there is already so much chaos in the system. In essence, the system is deemed to converge because it doesn't get any better nor any worse.

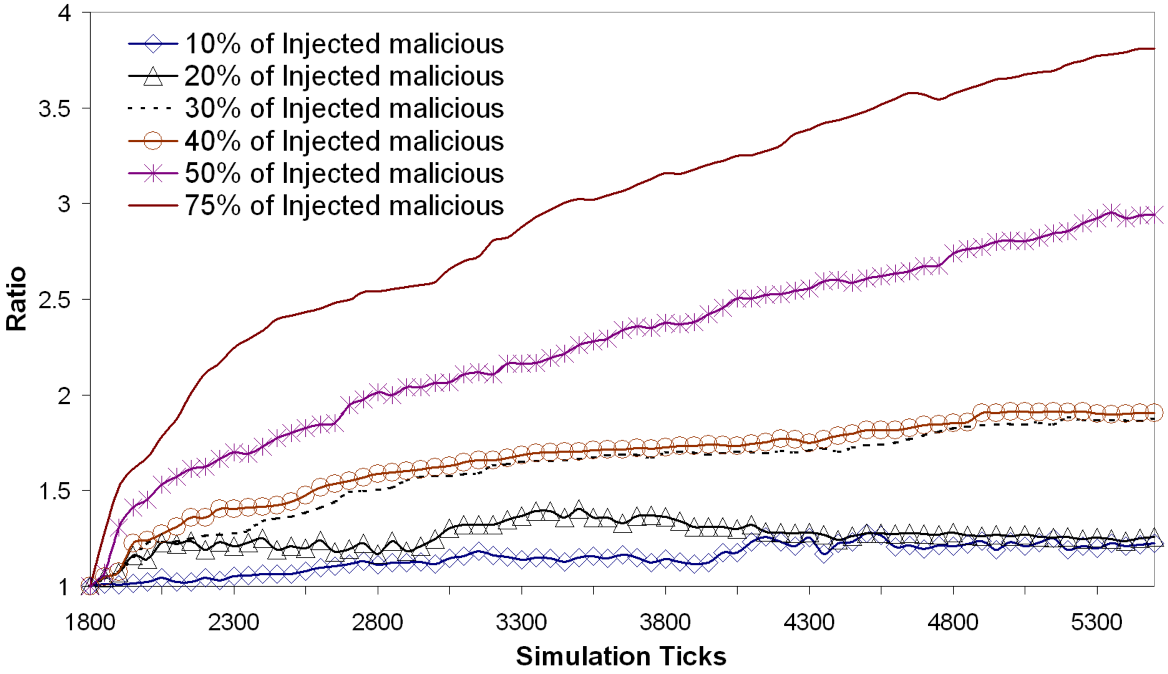


Figure 3.2: Injection of Disorder Attackers on Vivaldi: average relative error ratio.

Figure 3.3 shows the cumulative distribution of the relative error of the victims of an injected disorder attack. We clearly see that from 30% of malicious nodes the impact on the system can be considered as very serious with many nodes seeing a large increase in their relative errors. For a proportion of 50% or more malicious nodes, the system collapses with over 35% of the honest nodes computing coordinates that are

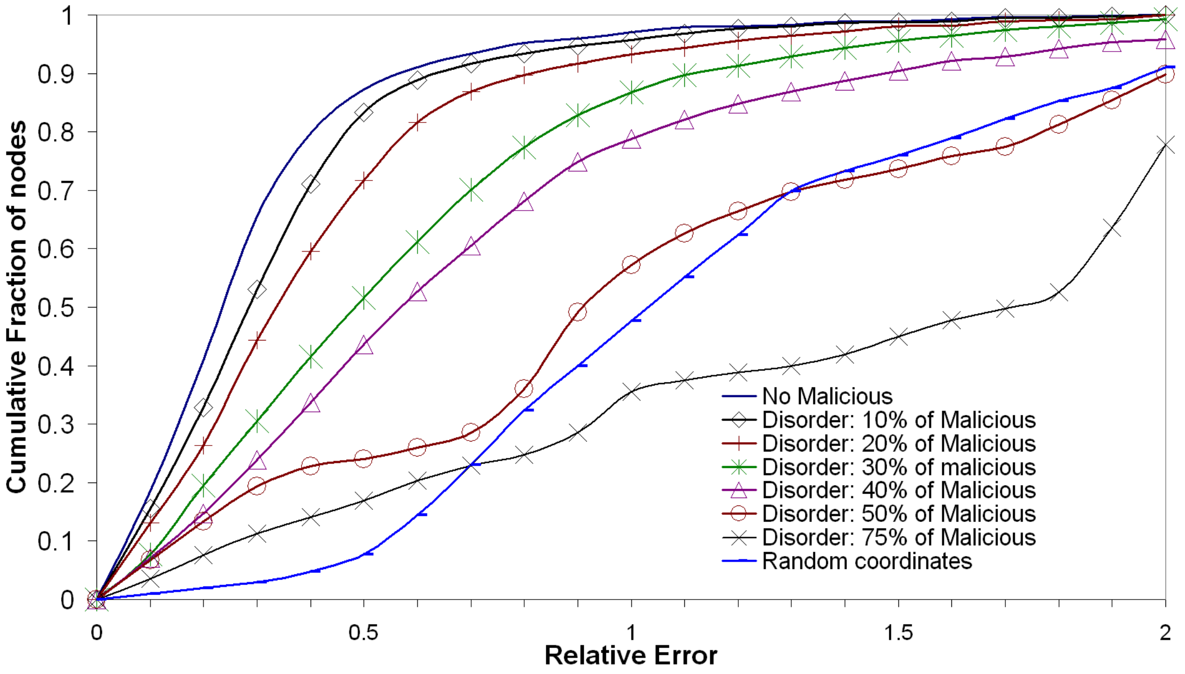


Figure 3.3: Injected Disorder attack on Vivaldi: CDF of relative error at simulation tick = 5000

similar or worse than if chosen randomly.

Figure 3.4 represents the impact of the space dimension on the attack. In this figure, the average relative error of honest nodes is measured after re-convergence. We see that the more accurate the Vivaldi system is in the absence of malicious nodes, the more vulnerable it is to the disorder attack. This is because the variation of more coordinate components for a point in a larger space results in higher displacement in that space. This observation is compounded for the 2-dimensional space augmented by a height as a variation of the height yields a greater effect on the node displacement. We also observe that in most cases, Vivaldi with half the population of malicious nodes is worse than a random coordinate system.

Figure 3.5 shows the impact of the attack as a function of the system size as measured a long time after the attack started. We see that a larger system is more difficult to impact for a same proportion of attackers. This is consistent with the fact that a larger Vivaldi system is more accurate, but also establishes that Vivaldi finds increased strength in a larger group. Put simply, this is because as one increases the number of springs in the system, the energy needed to disrupt it is higher. In our case, a larger group means more "good" forces to counteract and dissipate the effect of the malicious

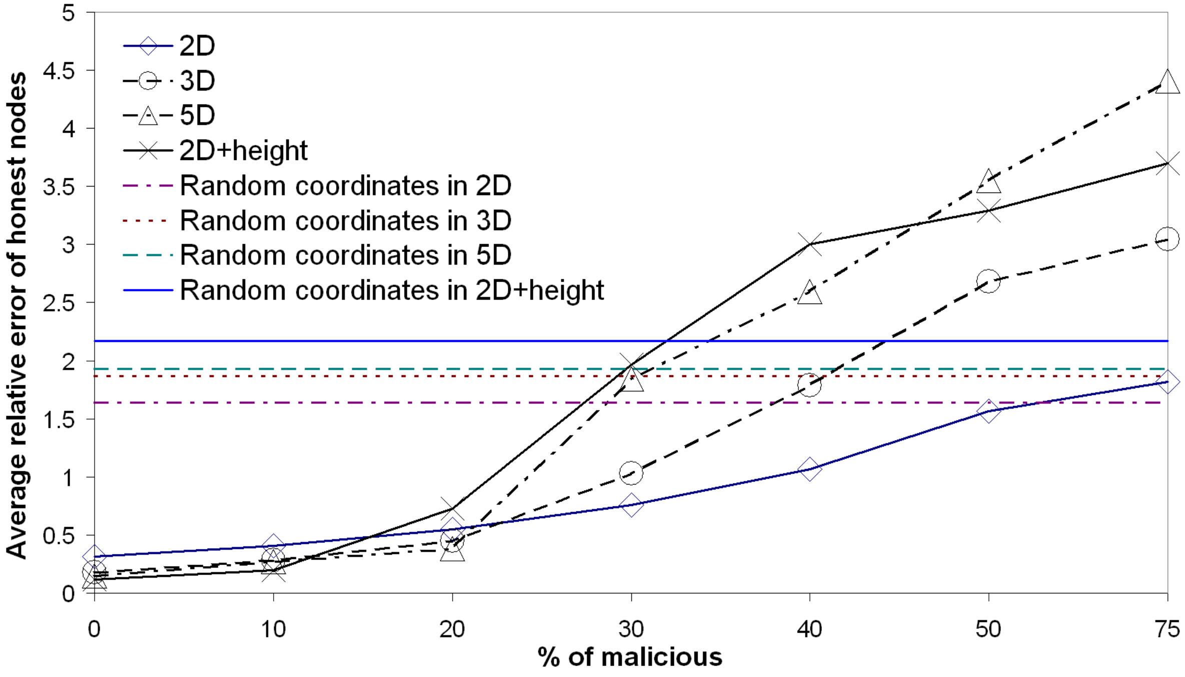


Figure 3.4: Injected Disorder Attack on Vivaldi: Impact of space dimensions

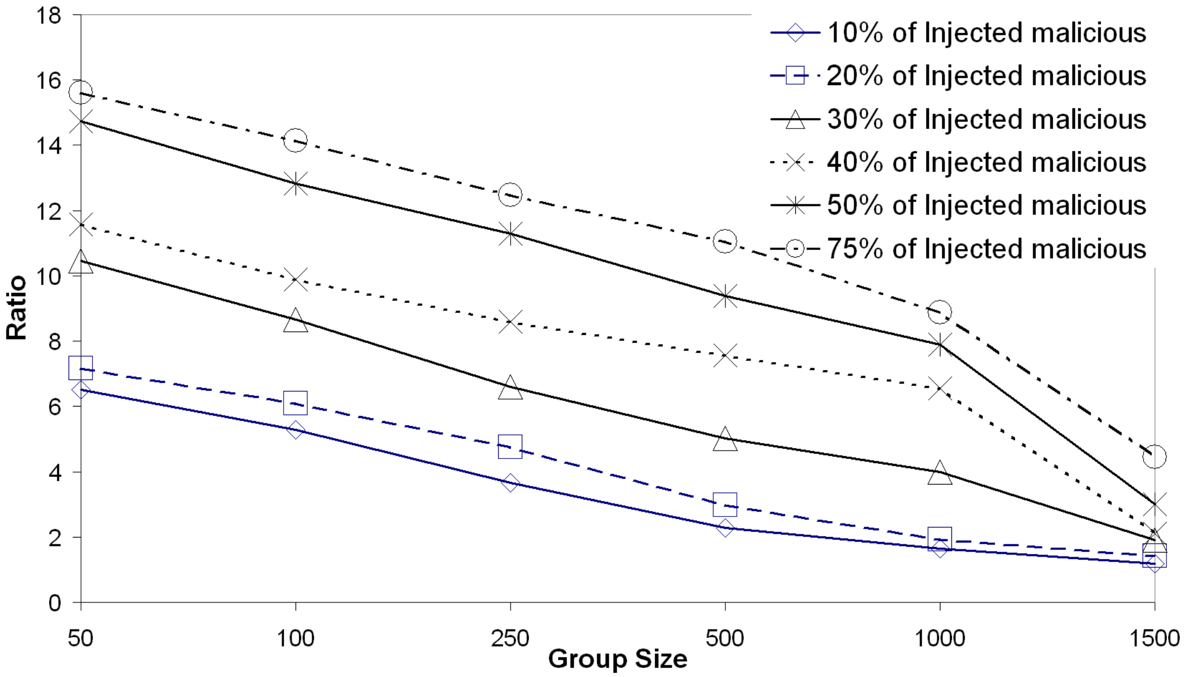


Figure 3.5: Injection of Disorder Attackers on Vivaldi: Impact of system size on the attack.

ones.

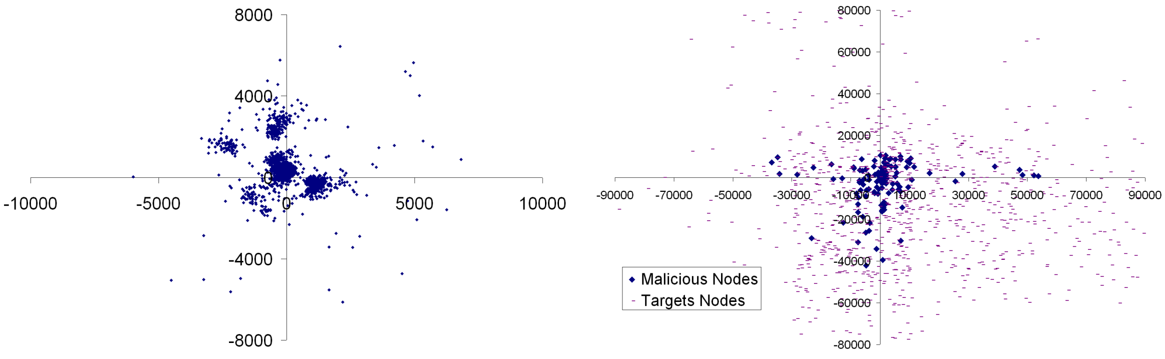
3.4.2 Repulsion Attack

In this scenario, malicious nodes are trying to isolate some nodes in the network, either by repulsing a set of targets away from other nodes in the coordinate space, or by repulsing all requesting nodes away from a selected target. The first attack consists in fixing coordinates where to isolate all requesting nodes, say X_{target}. It is important to notice that this value is set high enough to allow lie consistency. This means that the predicted distance after the lie should be equal to the measured distance. In fact, since we assume that a malicious node cannot shorten a distance measurement, but can however delay it, we must set the coordinates of both the victim and the malicious node to be consistent with this fact. Although for most network positioning systems, application probes are used, for generality purposes we design and test the attacks assuming ICMP ping probes. We assume here that malicious nodes know the current coordinates of their targets, $X_{Current}$, by means of previous requests for example. Malicious nodes are then able to compute the needed RTT that is consistent with the lie,

$$RTT = (\parallel X_{target} - X_{Current} \parallel /\delta) + \parallel X_{target} - X_{Current} \parallel$$

and to delay the measured RTT by:
$RTT_{needed} - 2 \cdot (ReceivedTimestamp - SendTimestamp)$. Each malicious node is selecting a random coordinate that is far away from the origin.

The effect of this attack can be visualized in figure 3.6, for our full-size system and



(a) Vivaldi (b) Vivaldi with 10% malicious

Figure 3.6: *Effect of Repulsion attack on Vivaldi with 1740 nodes* after convergence of the normal Vivaldi system. In this version of the attack, each malicious node sets a target coordinate independently for every honest node.

Figure 3.7 shows the cumulative distribution function of the measured average relative error after convergence in an repulsion attack. The gentler slope of the curves indicates that the impact of this type of attack is greater than in the case of a disorder attack (see fig. 3.3). This is because a repulsion attack is more structured and more consistent than a disorder attack, since the chosen target coordinate is always the same for every victim-attacker pair.

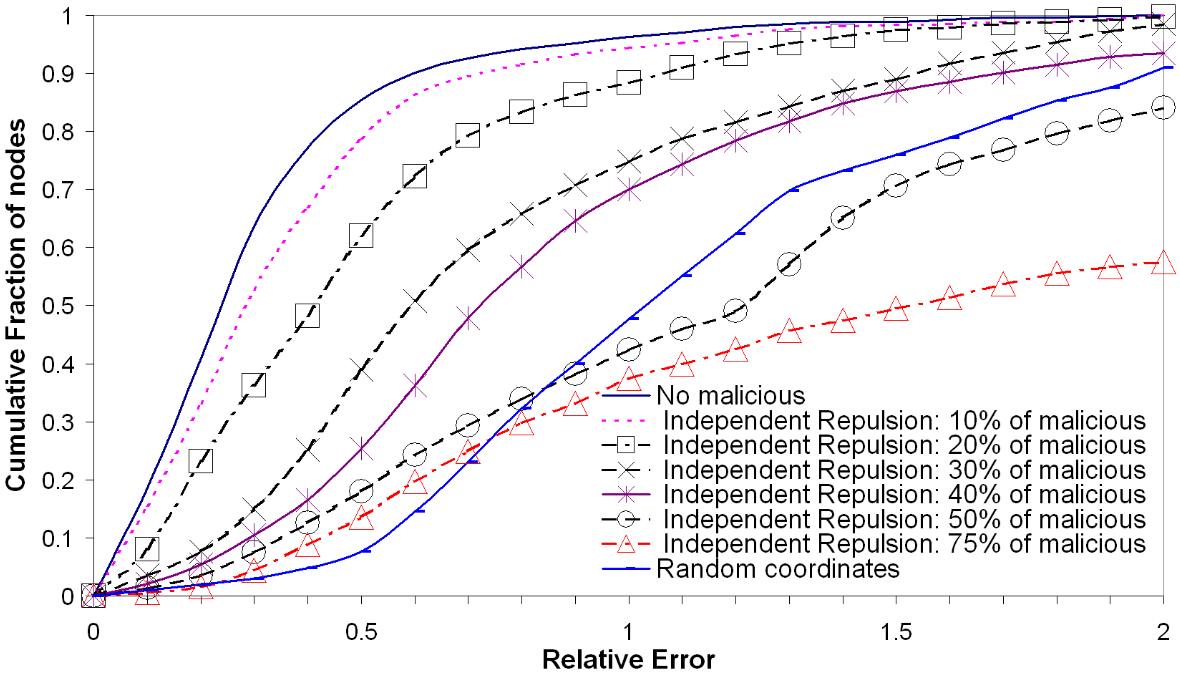


Figure 3.7: Injected Repulsion Attack on Vivaldi: CDF of relative error.

We study the effect of space dimension on this attack in figure 3.8. Again, the results confirm that the more accurate the system is without malicious nodes, the more vulnerable it is to attacks, which highlights a fundamental trade-off between accuracy and vulnerability.

So far, the repulsion attack consisted in each attacker attacking every other node. Figure 3.9 shows the effect of a modified repulsion attack where each attacker independently attacks a subset of the other nodes. Each attacker chooses its own target subset independently, along with their target coordinate values. However, the target subset size is fixed and equal for all attackers. We see that small subsets chosen in-

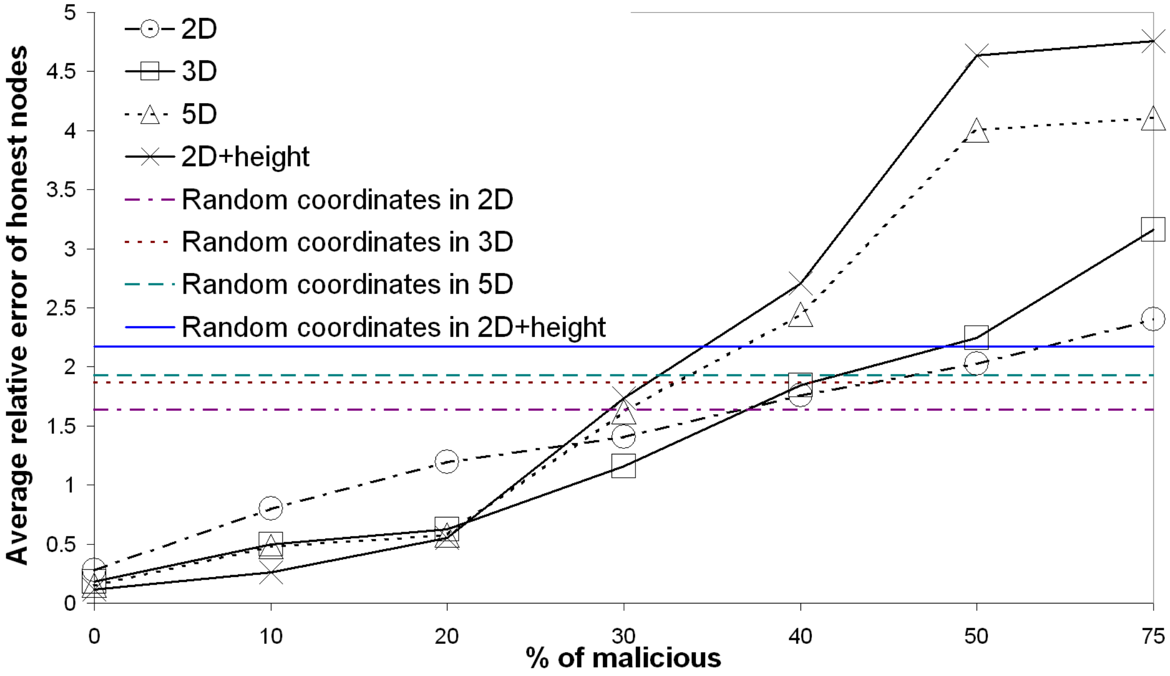


Figure 3.8: Injected Repulsion Attack on Vivaldi: impact of space dimensions.

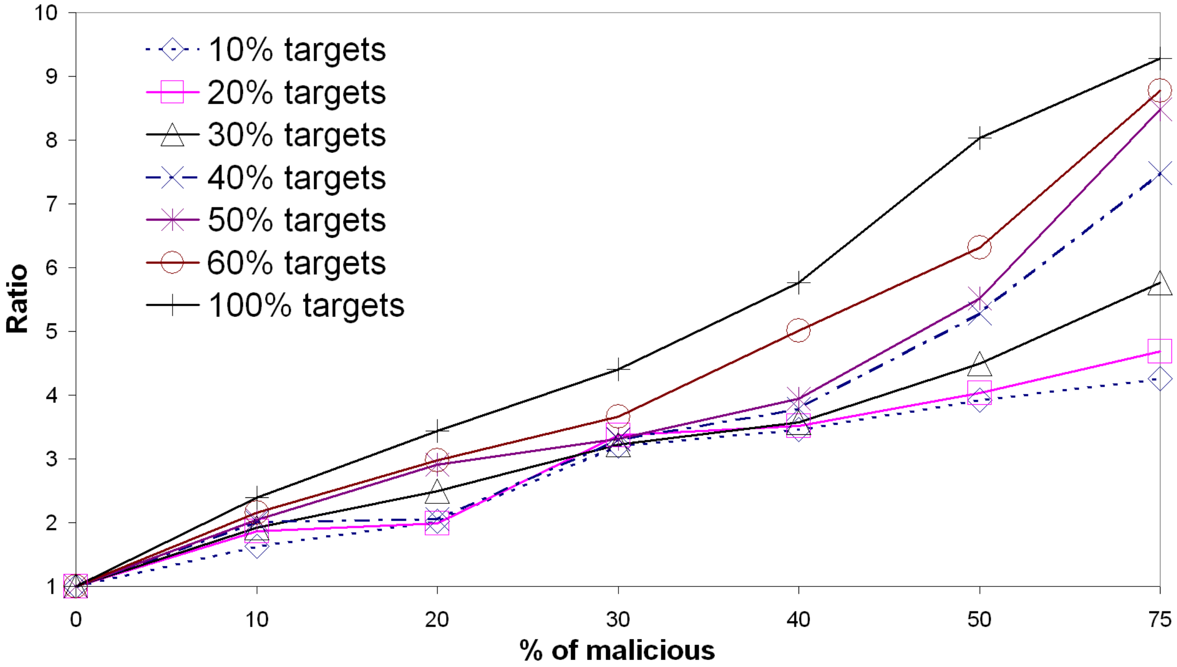


Figure 3.9: Injected Repulsion Attack on subsets of target nodes.

dependently result in a less effective attack and that there is no great difference in effectiveness when the set of attackers constitutes less than 30% the population. This can be explained by the fact that in such conditions the attack gets “diluted”, giving the system plenty of opportunity to correct itself through nodes that are under no, or very little, attack.

Figure 3.10 shows the response of a system under injection repulsion attack as a function of system size. As in the case of a disorder attack, larger systems reduce the impact of the attack. However, because a repulsion attack is much more consistent than a disorder attack, the system is less effective at countering the effects. This is why we observe higher values for the average relative error and a much gentler slope of the curve than in figure 3.5.

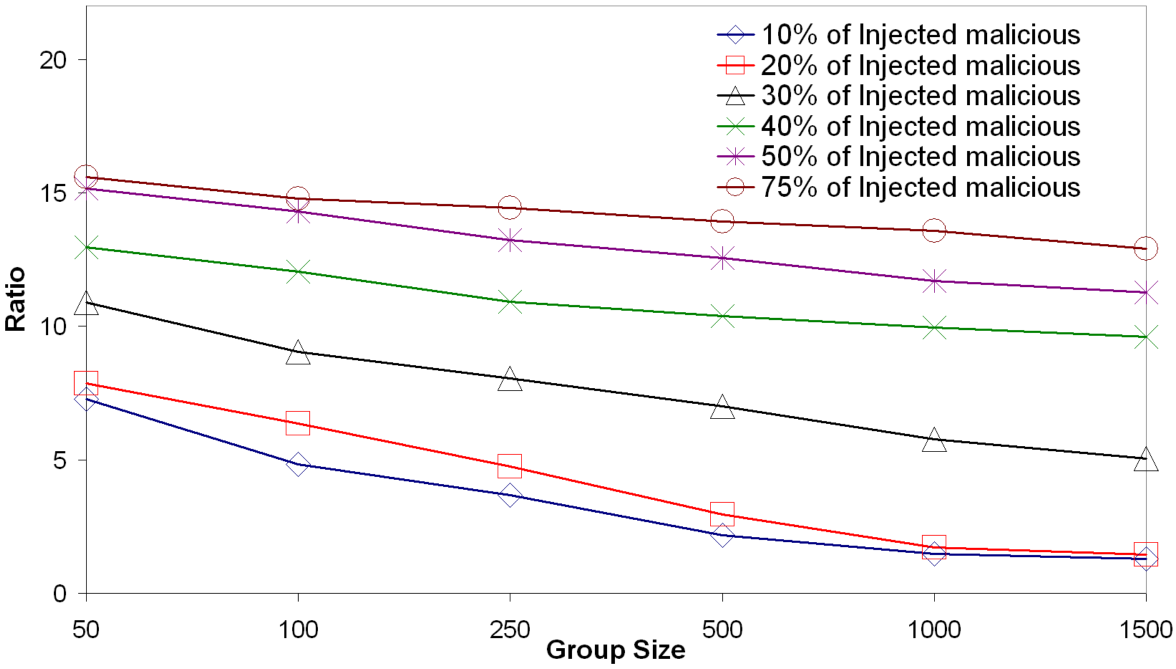
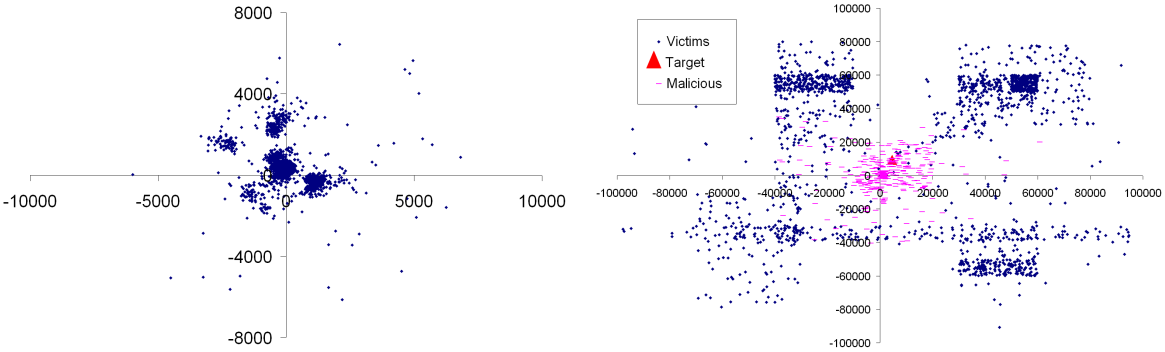


Figure 3.10: Injection Repulsion Attack on Vivaldi: effect of system size

3.4.3 Colluding Isolation Attack

This is a repulsion attack, aiming to isolate a particular target, where the attackers behave consistently in a collective way. They could, for instance, try and move all honest nodes consistently away from a same designated target node (strategy 1). That is, they agree on a distance from the chosen node for each victim and collectively and consistently direct victims towards their designated coordinate. Figure 3.11 shows the effects for such an attack. In this first scenario of the colluding isolation attack, we observe that the "cluster" structure originally present in figure 3.11(a), disappears in the presence of only 10% of malicious nodes. Honest nodes are moved away from their correct positions to the borders of the coordinate space, and malicious nodes keep their positions "around" the target node.



(a) Vivaldi

(b) Vivaldi with 10% malicious

Figure 3.11: *Effect of Colluding Isolation attack on Vivaldi with 1740 nodes*

Figure 3.12 depicts the effects of a colluding isolation attack on the system. The salient result is that the system can quickly become worse than a random coordinate system. Indeed, from 30% of malicious nodes in the system, the accuracy becomes equal or worse than if nodes chose their coordinates at random. This clearly demonstrates that colluding attacks are very potent due to their better structure and can have a great adverse impact on overall system performance.

Another strategy (strategy 2) of colluding isolation attack is for the attackers to set their coordinates in a remote area of the coordinate space (so that they are clustered in that area) and then to choose a victim target node and convince it that its own coordinate is within the attacker cluster. The target coordinate is set before the attack begins and agreed by all attackers.

We observe in figure 3.13 the variation of the relative error of the target node through time. We see that the first scenario of colluding isolation attack (consisting in repelling all other honest nodes from a chosen target) is more effective than trying to lure a target into a remote area of the space. Intuitively, this is because much more error is introduced in the system when more nodes are pushed away from their

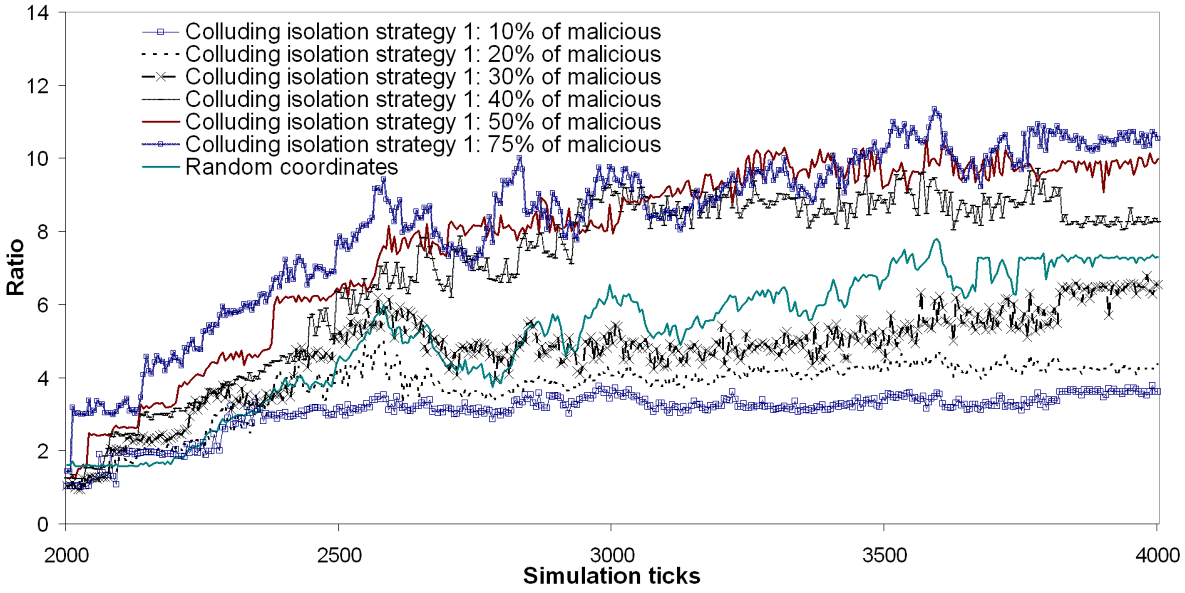


Figure 3.12: Colluding isolation Attack on Vivaldi: average relative error ratio

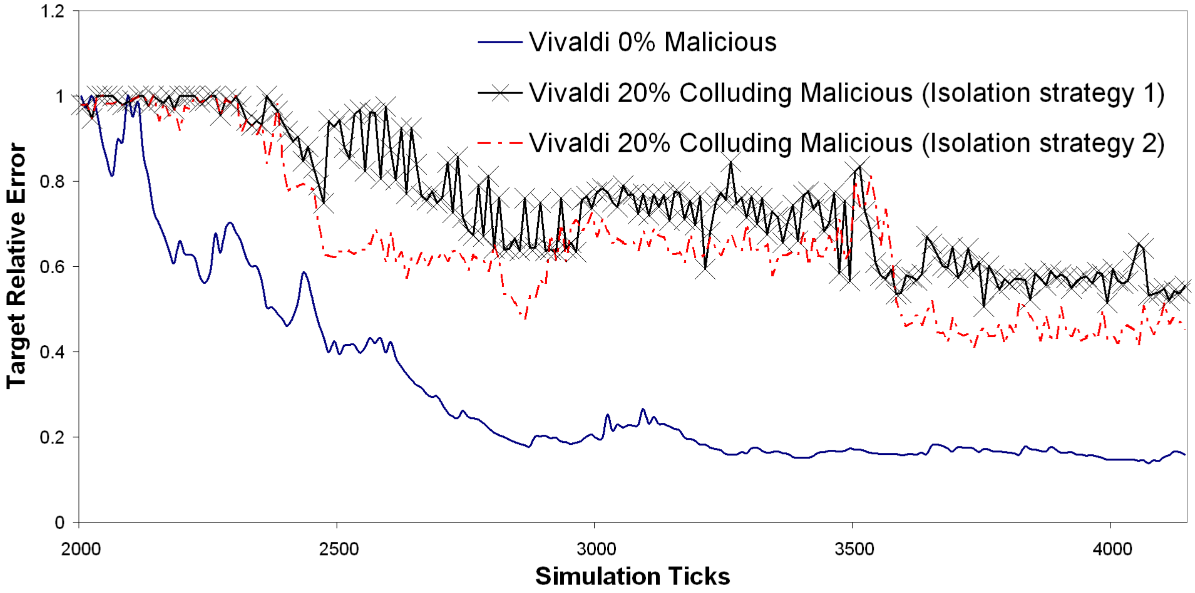


Figure 3.13: Colluding Isolation attack on Vivaldi: relative error of the target node correct position, thus resulting in more distortion of the coordinate space with greater repercussion on the final position of the target nodes. This is indeed confirmed by the results of figure 3.14 that depicts the cumulative relative error for the nodes in the system under both types of colluding attacks.

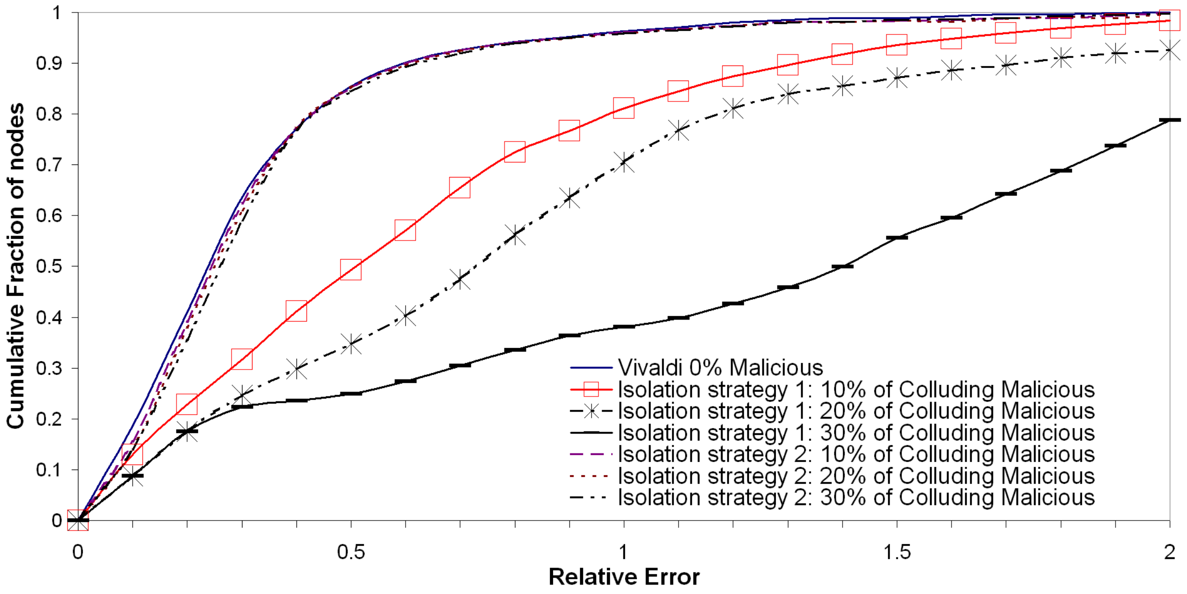


Figure 3.14: Colluding Isolation Attack on Vivaldi: CDF of relative errors.

3.4.4 Combined Attacks

In the context of system offering an always-on and large scale coordinate service, it is plausible to assume a constant and permanent low level of malicious nodes. Indeed, in the previous sections we have examined the effects of attack outbreaks. But in the wild, as has already been observed after major worm outbreaks and security warnings, once an outbreak has been contained and resolved, one can expect that some small portion of the systems are not upgraded for a very long time after the release of the necessary patches. This is especially true in the case of systems that are under many different administrative controls (as is the case for home personal computers). Figure 3.15 shows the impact of such low level combined attacks on Vivaldi, where colluding nodes implement strategy 1 of the colluding isolation attack. In these combined attacks, the percentage of malicious nodes of each type is the same. This figure shows

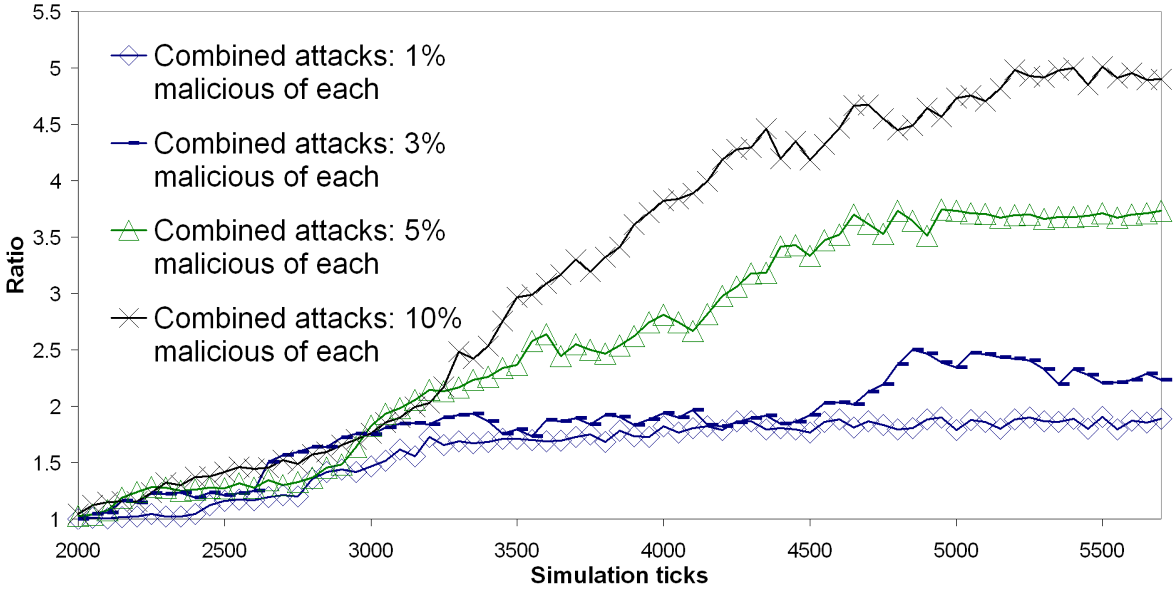


Figure 3.15: Combining attacks on Vivaldi (Disorder, Repulsion and Colluding Isolation attack strategy 1): impact on convergence.

that fairly low level of malicious nodes can still have a sizeable impact on the overall system performance, which, in turn, indicates that return to normality after an attack may take an extremely long time, if at all possible.

Finally, figure 3.16 confirms that larger systems are more resilient and recover better than smaller ones.

3.5 Attacks on NPS

Recall from the previous chapter, in section 2.3.4 that the NPS system includes a strategy for mitigating the effects of malicious attacks. Therefore, we experimented with NPS in both a secure and non secure version. Unless stated otherwise, the security mechanism is switched on. Note also that we consider the ideal, hypothetical case where the landmarks are highly secure machines that never cheat. The results we present in the following sections can therefore be considered as best case scenarios from a security point-of-view, as the impact of attacks could be much more severe should our security of landmark hypothesis not hold.

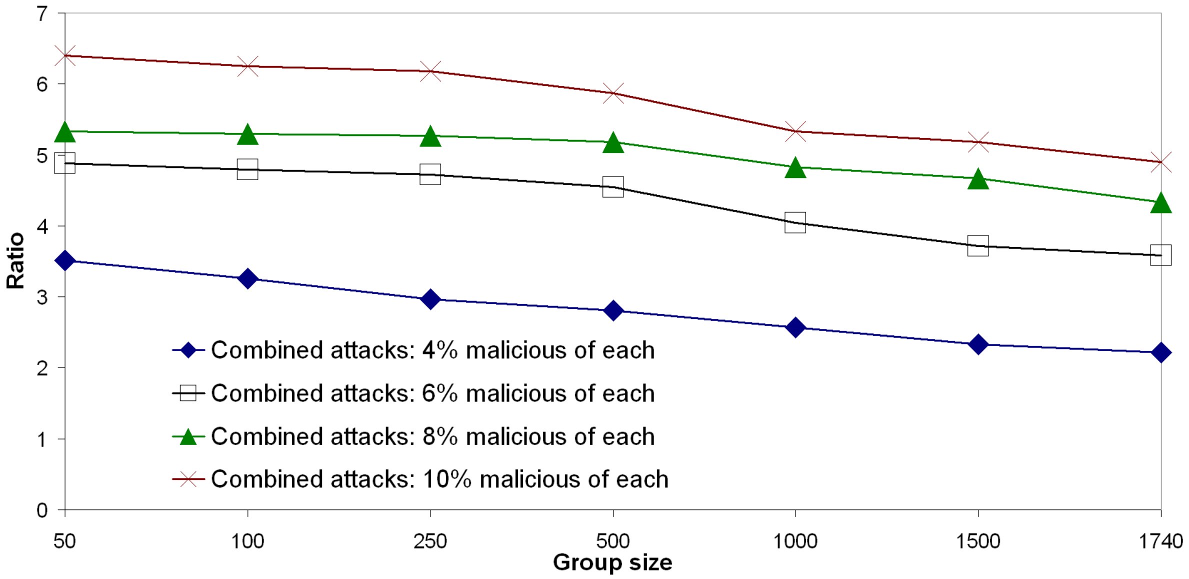


Figure 3.16: Combined attacks on Vivaldi: effect of system size.

3.5.1 Injection of Independent Disorder Attackers

In this first attack, when malicious nodes are chosen as reference points by the membership server (or when an already active reference point gets infected by malware), they perform simple attack that consists in transmitting the correct coordinates of the (malicious) reference point to the victim, and delaying measurement probes without caring about lie consistency. Figure 3.17 depicts the average relative error variation in function of time, while injecting after convergence of the system, a percentage of malicious nodes. When the malicious reference node detection mechanism is off, we notice the sharp climb in relative error when 20% of malicious nodes join the system. The accuracy of NPS is destroyed when cheating nodes get introduced in layer-1 of the measurement hierarchy. On the other hand, the malicious reference node detection mechanism is shown to be highly effective in combating such a malicious population of up to more than 30% of the overall population. However, a population of 40% or more malicious nodes in the system defeats the NPS security mechanism. This can be explained by the fact that the security mechanism relies on simple statistical properties of the observed errors (i.e the median) to filter out perceived outliers. In the presence of enough malicious nodes serving as reference points, the computation of the

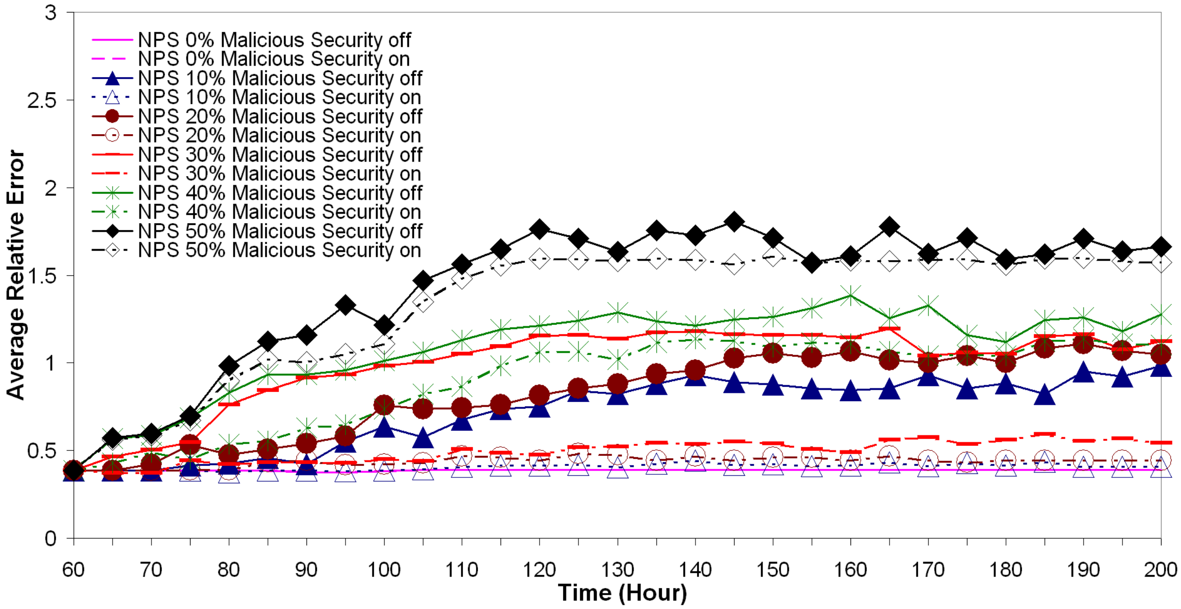


Figure 3.17: Injection in NPS of Independent Disorder attackers (No prevention): average relative error.

median itself gets skewed sufficiently that malicious behaviour is assimilated to normal behaviour. The cumulative distribution function of the measured average relative error shown in figure 3.18 confirms previous results. The gentler slope, and heavy tail feature, of the 40% and 50% curves when security is on indicates the impact of the attack when enough malicious nodes are introduced in the system. We observe that when introducing 40% of malicious nodes, only 50% of honest nodes would re-converge to a relative error less than 0.5.

Figure 3.19 shows the effect of space dimension when NPS is subjected to a simple disorder attack. Just as in the Vivaldi case, this experiment proves again that the more accurate the system is without malicious nodes, the more vulnerable to attacks it is. In particular, we observe that with more dimensions used in the coordinate space, the NPS system is much more vulnerable to a smaller portion of malicious nodes. We observe that systems running with 6 and 8 dimensions still can prevent against a minority of malicious nodes, whereas a simple attack can destabilize a 10 or 12-dimensions NPS system more easily. In the later cases, when malicious nodes only constitute 20% of the population, the relative error climbs to more than 1. From 50% of malicious nodes injected in the system, the accuracy becomes equal or worse than if nodes chose their

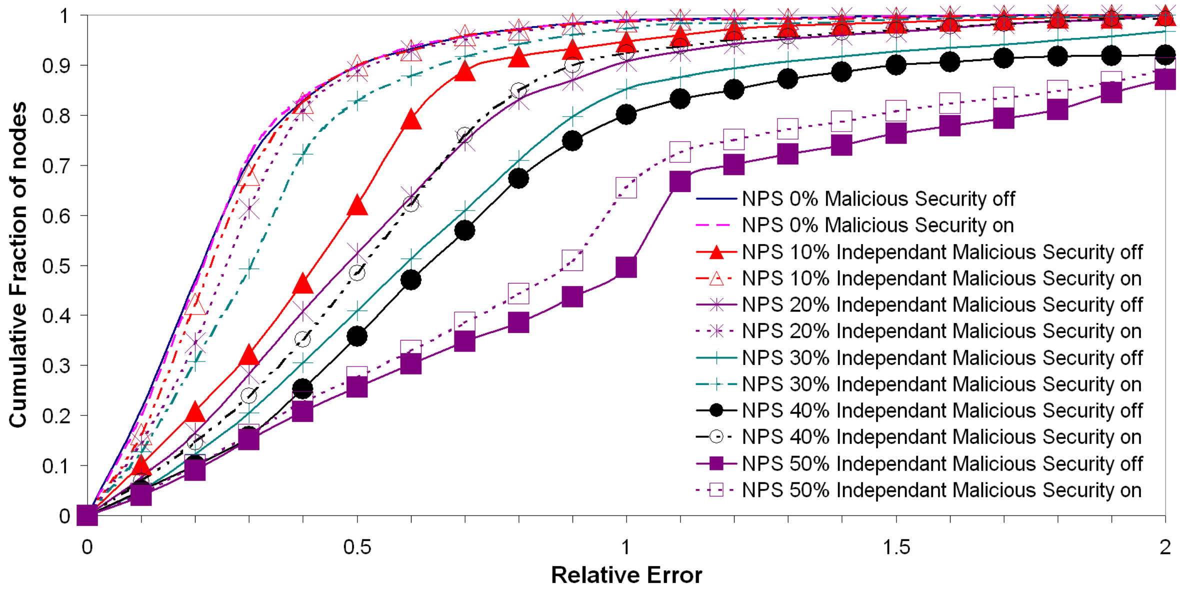


Figure 3.18: Injection in NPS of Independent Disorder attackers: CDF.

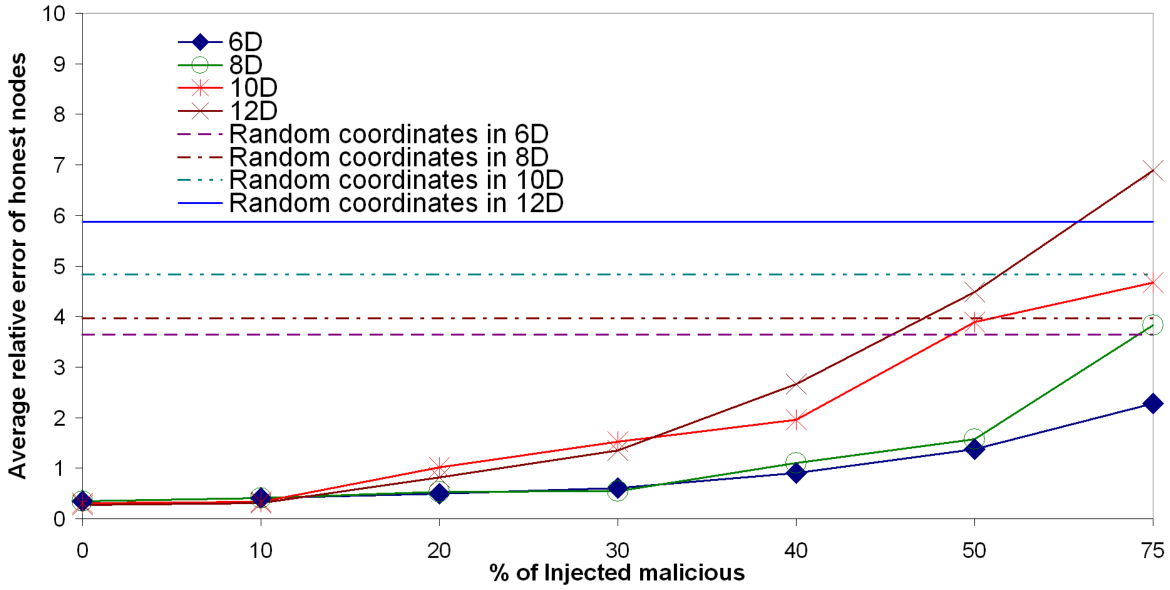


Figure 3.19: Injection of Independent Disorder attackers: Impact of dimensionality.

coordinates at random. This is explained by the fact that the more dimensions are used, the more "chances" malicious nodes get to become reference nodes, creating greater confusion among the honest nodes that depend on them in the layers below. Moreover, as in the Vivaldi case, more dimensions result in greater displacement in the coordinate space for the victim.

3.5.2 Injection of Naive Anti-Detection Disorder Attackers

In this section, we consider an attack whose primary objective is to try and defeat the NPS security mechanism, described in chapter 2 section 2.3.4. To this end, attackers will lie consistently about their position and inflate network distances by that corresponding amount, while paying particular attention that the relative error computed by the victim is lower than 0.01. Doing so essentially negates the very first condition checked to detect malicious nodes (see section 2.3.4 in chapter 2), in effect shutting down detection of the attackers.

First, we consider that malicious nodes know their targets' coordinates with a probability $p = 1/2$. We discuss next the effect of coordinate information on the efficiency of the attack. The target coordinate information allows first to better estimate the distance between the target and the attacker and second to compute the direction defined in the coordinate space by the nodes themselves. When not available, the malicious node sets a random direction and estimates the distance between itself and the target as $ReceivedTimestamp - SendTimestamp$.

As illustrated in figure 3.20, the attack consists in delaying the victims' probes by

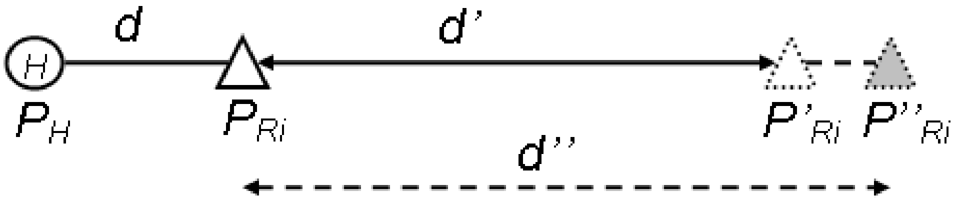


Figure 3.20: Anti-Detection NPS attack

$\| P'_{Ri} - P_{Ri} \| = d'$ such that

$$\| P'_{Ri} - P_{Ri} \| \gg d$$

and then send coordinates P''_{Ri} such that

$$\| P''_{Ri} - P_{Ri} \| < 0.01 \| P'_{Ri} - P_{Ri} \|$$

It is easily shown that

$$E_{Ri} < 0.01 \Rightarrow d'' > \frac{\alpha + 1.99}{0.01} \cdot d$$

with $\alpha\, d = d'' - d'$.

To make the attack harder and make the security mechanism of NPS behave in more realistic way, we add a probe threshold condition to each probe, such that a probe would be considered by the requesting node as suspicious if the RTT it measured was above that threshold. Such probes are then discarded. In the following simulations, the probe threshold is set to 5 seconds. In a first scenario, we consider malicious nodes that ignore this probe threshold, yielding a so-called naive anti-detection disorder attack.

In figure 3.21, we observe the average relative error variation after injection of malicious nodes in a converged NPS system. We see that this attack has a bigger impact on the whole system than the simple disorder attack (see figure 3.17), causing greater average relative errors. We also observe that the attack is very effective at defeating the security mechanism, with the security-protected relative errors only trailing marginally

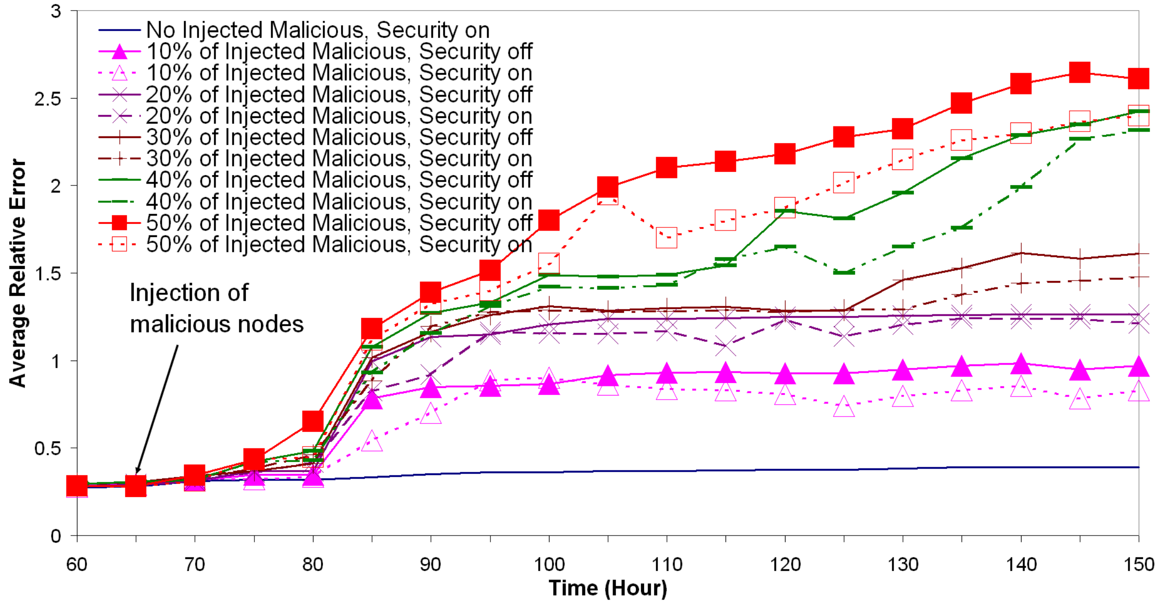


Figure 3.21: Injection in NPS of Anti-detection naive attackers: impact on convergence.

the errors observed when no security mechanism but the probe threshold is employed. This is despite the attacker guessing half of the time, and could therefore appear surprising. However, the reader should note that the NPS security mechanism discards at most one malicious reference point at each positioning (i.e. the one yielding the greater error), giving the malicious nodes potentially several reprieves on bad guesses.

As for the Vivaldi system, we note that in presence of only a minority of malicious nodes, despite the system converging in the sense that the relative errors at each node stabilize, these errors are so high that a great variation of the coordinates does not affect the associated error.

We measured the impact of dimensionality and group size on the effectiveness of this attack and found the now expected results that higher precision (i.e. higher dimensionality) was more affected while larger groups present a better immunity.

More interesting in this attack is the effect the knowledge of the attacker has on its effectiveness. In figure 3.22, we show the relative error ratio for various probabilities that the attacker knows a victims's coordinates prior to striking. We see that in the presence of a small malicious population, full knowledge of victims' coordinate can almost triple the effectiveness of the attack compared to the pure guess work case. However,

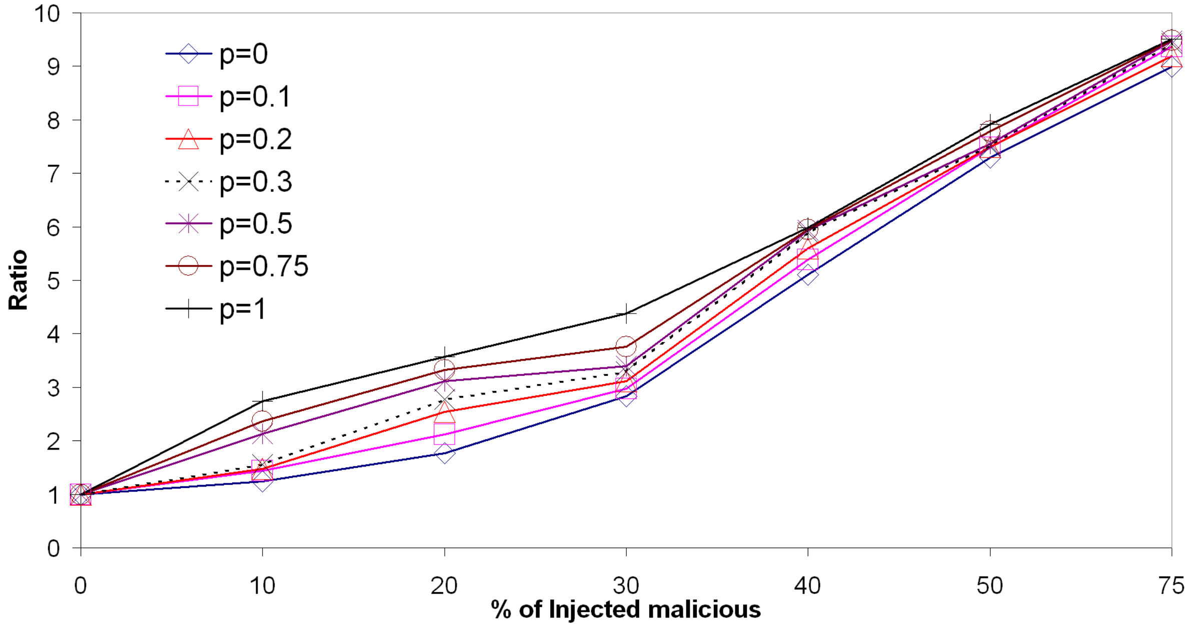


Figure 3.22: Injection in NPS of Anti-detection naive attackers: effect of victims coordinates knowledge.

as the population of malicious nodes grows, the benefits of more knowledge diminish. This confirms again that, regardless of the sophistication of the attack, the NPS security system soon gets overwhelmed when the population of malicious node exhibits a certain critical mass. As figure 3.23 shows by representing the ratio of malicious nodes filtered to the overall number of filtered nodes by the security mechanism, this critical mass is about 20% (about half the needed population of malicious node compared to the simple disorder attack). Furthermore, this figure also confirms that, as more and more malicious nodes are able to operate in all impunity, the errors they introduce in the positioning of honest nodes result in higher false positive rates with the security mechanism filtering out more and more (mis-positioned) honest reference points. But because at most one reference point gets filtered per positioning, these false positives actually create some extra protection for the malicious ones.

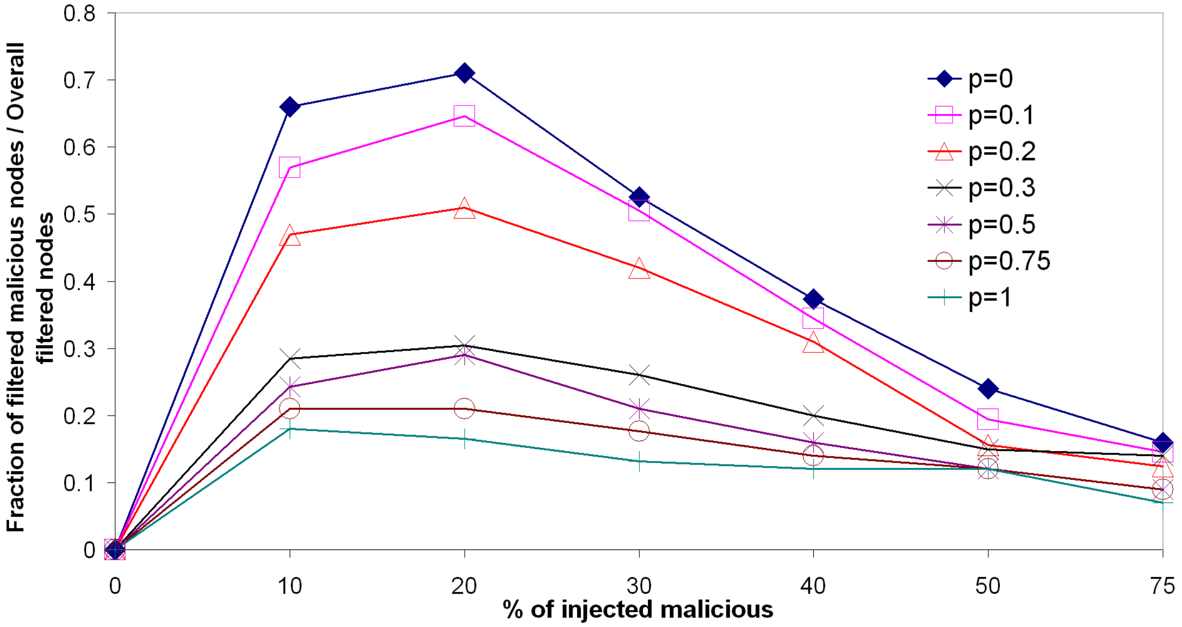


Figure 3.23: Injection in NPS of Anti-detection naive attackers: effect of victims coordinates knowledge on the ratio of filtered malicious nodes over the overall filtered nodes.

3.5.3 Injection of Sophisticated Anti-Detection Disorder Attackers

We now present a modification of the previous attack where the malicious nodes make an attempt to not only defeat the NPS security mechanism but also avoid detection by the probe threshold mechanism. To do so, an attacker will only interfere with the positioning process of nodes known, or believed, to be nearby. Indeed, if we recall the discussion in section 3.5.2, with a probe threshold of 5 s and $\alpha = 2$, then $d'' + d < 5s \Rightarrow d < 25ms$ in order to avoid detection by the NPS security system, d being the real distance between an attacker and its victim. As this attack is bound to be less detectable by the security mechanisms than the previous one which already yielded small differences between the ”security on” and ”security off” cases, only results in the presence of these security mechanisms are presented here. Unless stated otherwise, the attackers guess the position of their victims half of the time.

Figure 3.24 shows the cumulative distribution function of the relative errors in a

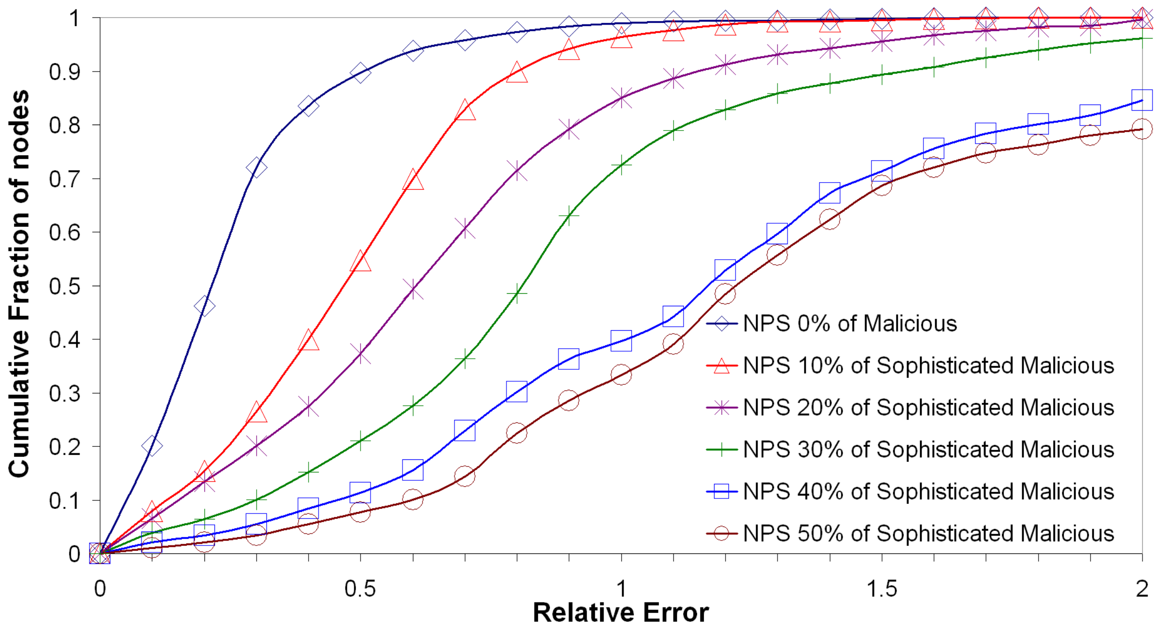


Figure 3.24: Injected Anti-detection Sophisticated attacks on NPS: CDF.

system under sophisticated anti-detection disorder attack. Clearly, this attack is devastating on the overall accuracy of the coordinate system, despite the attackers being more selective of their victims. This is because, even though the errors introduced by each attacker are smaller than in the naive case (nodes that are closer can only be ”pushed” less aggressively if the attacker is to avoid detection), these errors are allowed to permeate unchallenged through the system, propagating more widely through the undetected mis-positioning of honest nodes. We observed that in the system without malicious nodes, the mean relative error converged towards a value of about 0.4. Here we see that as little as 10% of attackers leaves over 60% of the overall population worse off than the average node in a clean system. We also observed that compared with the more naive version of this attack (figure 3.21), the sophisticated version induces higher average errors.

Again, better accuracy (i.e. higher dimensionality) and smaller group sizes were observed to be more sensitive to the attack.

Figure 3.25 shows the impact of the attacker's knowledge on the attack. By going from pure guessing to full knowledge (i.e. attacking only victims whose coordinates are known), an attacker can reduce by half its chances of being caught. We also see that the intrinsically more cautious strategy of this attack dramatically reduces the chances of an attacker being detected compared with the naive attack case (figure 3.23), especially when malicious nodes represent a smaller proportion of the population and operate with little knowledge of the exact coordinates of their victims. Indeed, for the case where the attackers never know exactly the coordinate of their victims, figure 3.25 shows that over 75% of all detections are false positives for attackers populations of 10% and over of the group.

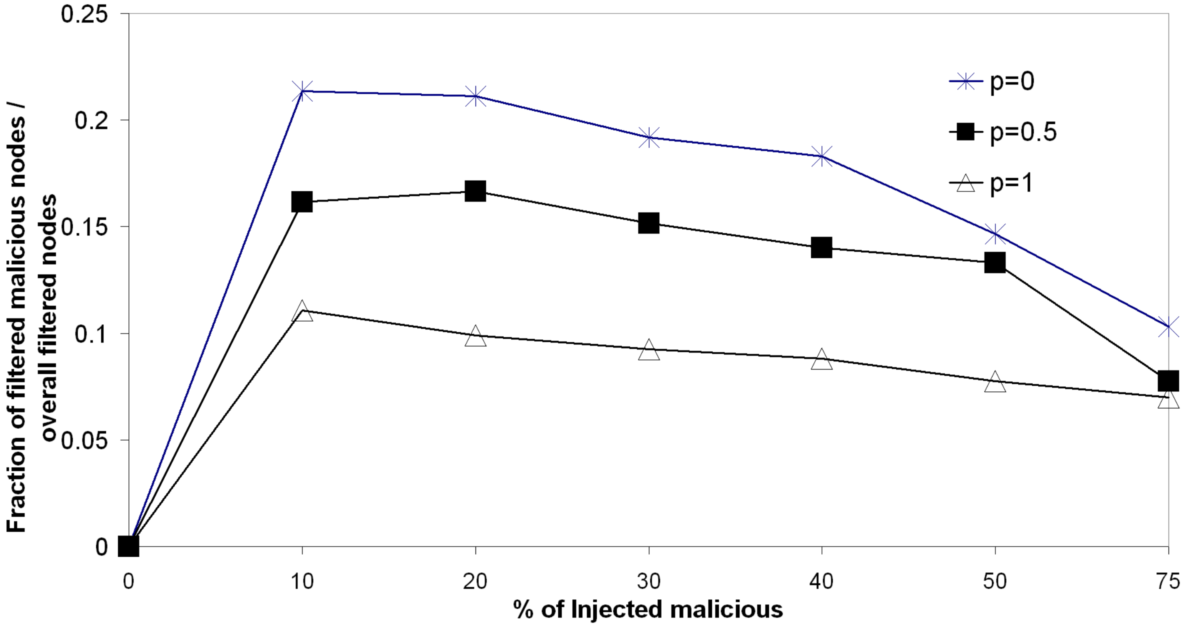


Figure 3.25: Injected Anti-detection Sophisticated attacks on NPS: effect of victims coordinates knowledge on the ratio of filtered malicious nodes over the overall filtered nodes.

3.5.4 Injection of Colluding Isolation Attackers

In a colluding isolation attack, the malicious nodes cooperate with each other and behave in a correct and honest way until enough of them become reference points at the same layer. Once at least a minimum number of malicious reference points has been reached (in our simulation this number is set to 5), these attackers identify a common set of victims. When involved in the positioning of any other nodes, the attackers do not cheat; while when dealing with a target node, they agree to pretend they are all clustered into a remote (far away) part of the coordinate space and carry out a naive anti-detection attack on the victim. The goal of this attack is to push the victims into a remote location at the ”opposite” of where the attackers pretend to be, thus isolation the victims from all the other nodes (in the coordinate space). The other main idea behind this attack is that by acting in a consistent way as a group, the attackers can maybe avoid detection by influencing the value of the median relative error (condition 2 of the NPS security mechanism – see 2.3.4). Also, as already mentioned, even if detected, at most one attacker would be filtered at each positioning, giving the others more opportunities to act.

We consider 2 scenarios for this attack. The first scenario consists in experimenting with a 3-layer NPS system, i.e. a system with the landmarks in layer-0, 20% of nodes serving as reference points in layer-1, and the rest of the nodes in layer-2. The second scenario is aimed at observing the propagation of errors through different layers and uses a 4-layer NPS system, with 2 layers (layer-1 and layer-2) containing 20% of the nodes acting as reference points.

Figures 3.26 and 3.27 show the cumulative distribution function of the relative errors in a 3-layer and 4-layer NPS system (respectively) under this colluding isolation attack. We observe a striking difference of impact depending on the structure of the NPS system. Indeed, the overall accuracy of a 3-layer system is much less affected than the accuracy of a 4-layer system. On the one hand, it is worth remembering that, in the 3-layer system, non victim nodes do not see any degradation of the accuracy of their positions (compared to a clean system), because they observe an honest behaviour from the attackers. This means that the overall degradation in accuracy is caused by the mis-positioning of the victims only. Hence, the perceived little impact of the attack depicted in figure 3.26 actually tends to indicates that the attack is very effective on the victim.

On the other hand, in a 4-layer system, some of the victims may be unwittingly se-

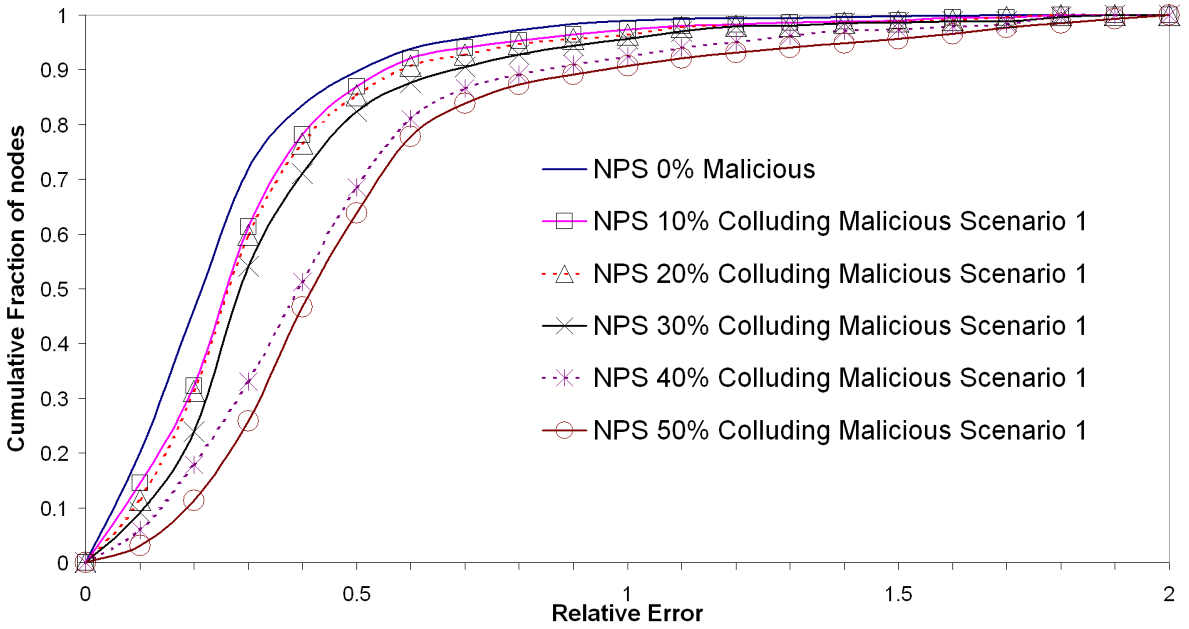


Figure 3.26: Injection of colluding Isolation attack on NPS in scenario 1: CDF of relative errors.

lected by the membership server to act as layer-2 reference points. The position errors inflicted on these nodes is then propagated through the rest of the system, resulting is an amplification of the errors from layer to layer. This is demonstrated in figure 3.28 that shows the average relative error of layer-2 and layer-3 nodes in clean 3-layer and 4-layer systems respectively, as well as the average relative error observed by layer-2 targets and layer-3 nodes in corrupted systems with a population of 20% of malicious nodes. From this figure, it is clear that the impact of layer-1 cheats on layer-2 victims is independent of the system structure (the curves are similar), layer-3 nodes of an attacked 4-layer system experience the worse mis-positionning. This propagation and amplification of the errors in this 4-layer system can be seen as a system-control attack (see section 3.2).

Finally, as in the Vivaldi case, we observe in figure 3.29 the impact of several small population of attackers which concurrently carry out all the previous attacks. This is reminiscent of a situation where some nodes are still misbehaving for some time following the release of patches and updates after a major outbreak of malware. Again, we see that attacks can have long lasting consequences on the operation of the coordinate system.

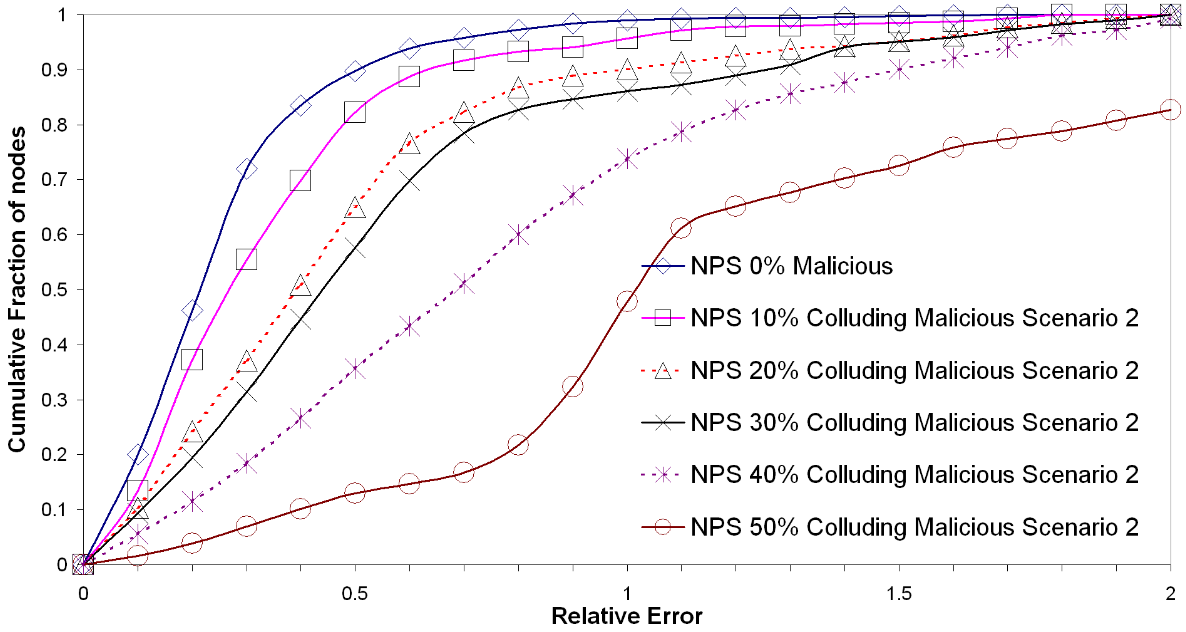


Figure 3.27: Injection of colluding Isolation attack on NPS in scenario 2: CDF of relative errors.

3.6 Conclusions

In this chapter, we have studied various types of attacks on two prominent coordinate system proposals. One of our salient findings is that larger systems are consistently more resilient than smaller ones. Given the observation in [15] and [14] that larger systems are more accurate and the well known fact that larger systems converge slower at start-up time, there seems to be a compelling case for large-scale coordinate systems to be built as a virtual infrastructure service component. The paradox is of course that always-on, large-scale systems supporting many different applications will always attract more attacks than systems with a smaller reach, while the large size of the system itself would act as a particularly good terrain to create especially virulent propagation of the attack.

Our results also show that there is an intrinsic trade-off to be made between accuracy and vulnerability. Indeed, we have shown that the more accurate the system for a given system size, the more susceptible it was to a same proportionate level of attack.

Also, we have shown that while an attack is in full swing, the performance of the coordinate systems (and of the applications it supports) can easily degrade below that of a systems where coordinates are chosen randomly, whilst the aftermath of an attack

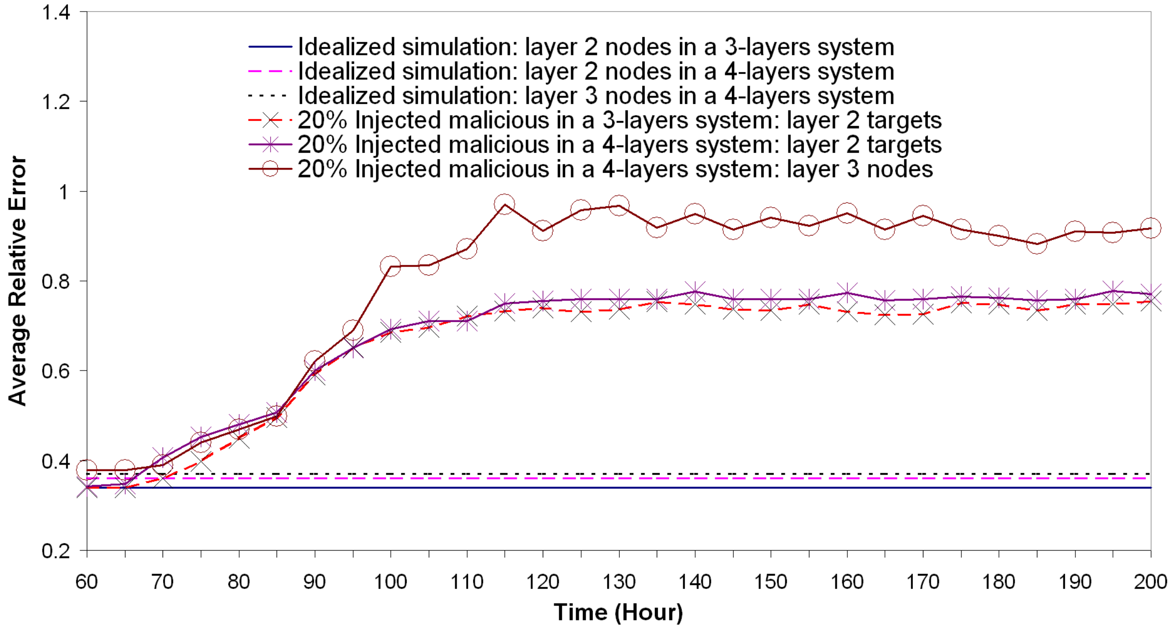


Figure 3.28: Injection of colluding Isolation attack on NPS: Propagation of errors.

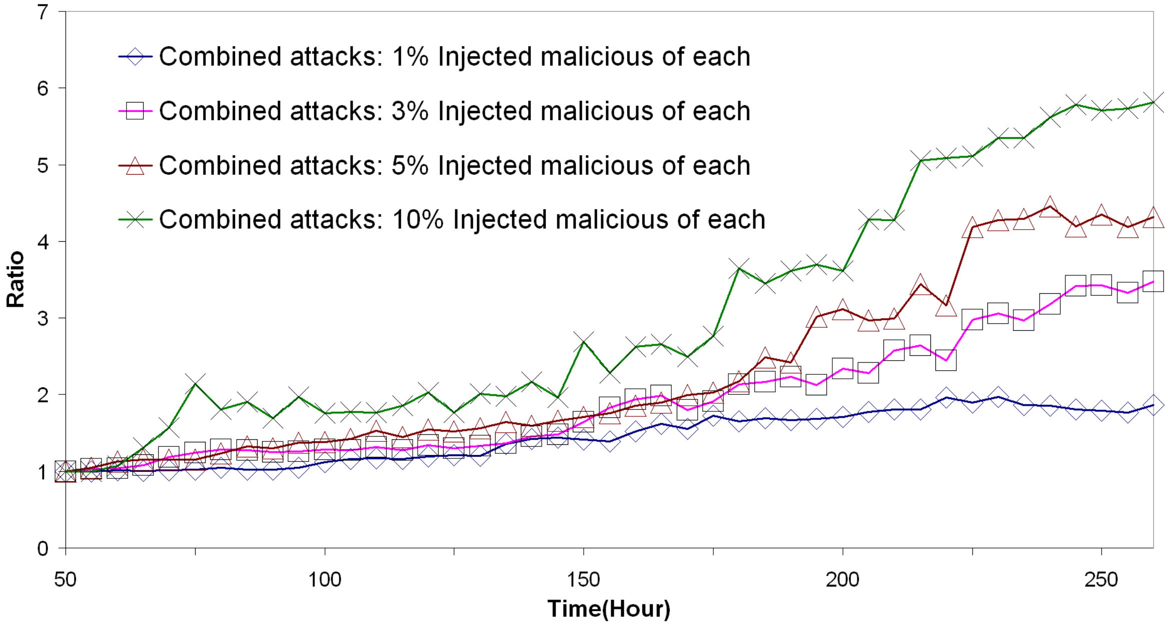


Figure 3.29: Injection of combined attacks on NPS (Independent disorder, Anti-Detection Sophisticated disorder and colluding isolation attackers): Impact on convergence.

could have very long lasting effects on the system due to a small number of remaining malicious nodes.

We have also shown that infrastructure-based systems can, under some well chosen attack strategies, be as vulnerable as those based on the peer-to-peer paradigm. Furthermore, the security mechanisms that have been proposed to date to defend against malicious nodes are clearly rather primitive and still in their infancy and definitely cannot defend against all types of attacks.

In the next chapter, we will concentrate on designing generic defense and security method to protect coordinate-based systems from large-scale malicious attacks. This work will be guided by the understanding of attack mechanisms and of their consequences on the coordinate systems gained from the study presented in this chapter. First, we should be able to propose a general detection protocol, that do not entail any change to the operations of the embedding systems. Second, our detection protocol should not be sensitive to the dimensionality used by the coordinate-based system, avoiding in this way any trade-off between accuracy and vulnerability. Finally, we should design a detection protocol that should be capable of surviving a potential very large range of sophisticated attacks, without relying on the geometric properties of the coordinate space.

4

Securing Coordinate Embedding Systems

4.1 Summary

This chapter addresses the security issues of the embedding phase in Internet Coordinate Systems, by proposing a general method for malicious behavior detection during coordinate computations. We first show that the dynamics of a node, in a coordinate system without abnormal or malicious behavior, can be modeled by a Linear State Space model and tracked by a Kalman filter. Then we show that the obtained model can be generalized in the sense that the parameters of a filter calibrated at a node can be used effectively to model and predict the dynamic behavior at another node, as long as the two nodes are not too far apart in the network. This leads to the proposal of a Surveyor infrastructure: Surveyor nodes are trusted, honest nodes that use each other exclusively to position themselves in the coordinate space, and are therefore immune to malicious behavior in the system. During their own coordinate embedding, other nodes can then use the filter parameters of a nearby Surveyor as a representation of normal, clean system behavior to detect and filter out abnormal or malicious activity. A combination of simulations and PlanetLab experiments are used to demonstrate the validity, generality, and effectiveness of the proposed approach for two representative coordinate embedding systems, namely Vivaldi and NPS.

4.2 Introduction

In the previous chapter, we have shown how coordinate-embedding services could be vulnerable to malicious attacks, providing a potentially attractive fertile ground for the disruption or collapse of the many applications and overlays that would use these services [2]. There are actually two obvious ways to disrupt the operation of a coordinate based system. First when requested to give its coordinate for a distance estimation at the application-level, a malicious node could simply and blatantly lie. Second, a malicious node, or even a colluding group, may aim at disrupting the embedding process itself. This latter strategy is very insidious and effective as it can result in important distortions of the coordinate space which then spoils the coordinate computations of many nodes (malicious and honest alike) [2]. This chapter focuses on developing and studying generic methods to secure the coordinate embedding process (the problem of nodes lying about their coordinate during distance estimation at the application-level is addressed in the next chapter).

More precisely, the embedding process, regardless of the actual coordinate-based positioning system, works on the premise that nodes adjust their coordinate based on some comparison between measured and estimated distances to some other nodes. Malicious nodes can interfere with this embedding process by, amongst other things, lying about their real coordinate and/or tampering with measurement probes, to create a discrepancy between measured and estimated latencies, so that unsuspecting nodes would wrongly adjust their own coordinate in a bid to reduce the difference [1]. Because the load on the network naturally varies in time, so does latency between pair of nodes, and as a result, the embedding process must be run periodically by all nodes to track changes in network conditions. This “continuous” adjustment of nodes’ coordinates can not only result in a drift of the coordinate space [18] but also gives plenty of scope and opportunities for malicious activity. We therefore seek to equip (honest) nodes with a means to detect, with low overhead, malicious activities they may encounter during embedding.

Noting that, in the absence of malicious nodes, a node’s coordinate depends on the combination of network conditions and the specificities of the embedding process itself (e.g. which coordinate protocol is in use, the chosen dimensionality of the geometric space, etc), we therefore introduce the concept of *Surveyor nodes* (or *Surveyors* in short). Surveyors form a group of trusted (honest) nodes, scattered across the network, which use each other *exclusively* to position themselves in the coordinate space.

Of course, Surveyors do assist other nodes in their positioning (as prescribed by the embedding protocol), but we stress that Surveyors never rely on non-Surveyor nodes to compute their own coordinate. This strategy thus allows Surveyors to experience and learn the natural evolution of the coordinate space, as observed by the evolution of their own coordinate, in the absence of malicious activities. In essence, Surveyor nodes are thus vintage points guaranteed to be immune from malicious activities. The idea is that Surveyors can then share a “representation” of normal behavior in the system with other nodes to enable them to detect and filter out abnormal behavior.

We postulate and verify that, in the absence of malicious activity, a node’s coordinate can be viewed as a stochastic process with linear dependencies whose evolution can be tracked by a Kalman filter [4]. Each Surveyor then computes and calibrates the parameters of a linear state space model and shares the parameters of this model with other nodes. These nodes can then use these parameters, to run locally and in a “stand-alone” fashion a Kalman filter tracking the coordinate adjustments. These nodes can then use the Kalman filter output (the innovation process), to compare their observed coordinate adjustments with the one predicted by the Kalman filter, and flag as “suspicious” embedding steps where the difference would be too high.

In section 4.3, we present a general model of coordinate embedding, in the absence of malicious nodes, that naturally leads to the Kalman filter framework. In section 4.4, we validate the model, with both simulations and PlanetLab experiments, in the case of both Vivaldi [15] and NPS [14]. This section also studies the viability of the idea of using Surveyor nodes in secure coordinate embedding. We then describe and evaluate, in sections 4.5 and 4.6, how Surveyors can effectively be used for malicious node detection in the specific embedding process of Vivaldi and NPS.

4.3 Coordinate Embedding Model

The goal of embedding systems, regardless of the embedding method and geometric space used, is to assign a coordinate to every node in the system so that, at any time, the distance between any two points in the geometric space should provide a good estimate of the network distance, measured as an RTT (Round Trip Time), between the corresponding nodes. Obviously, because at any instant in time, the RTT that can be measured between two nodes depends on the state of the network (e.g. traffic load, state of queues in routers, etc) as well as the state of the operating system in nodes (e.g.

scheduling state generating measurement noise, etc), the exact value of the RTT varies continuously. However, it has been shown that RTT values in the Internet exhibit some stability in a statistical sense [59], with the statistical properties of RTTs exhibiting no significant change at timescales of several minutes. It is that property that embedding systems exploit to provide good distance estimates while only needing to have nodes adjust (recalculate) their coordinate on a periodic basis. Consequently, the coordinate of a node can be viewed as a discrete stochastic process, and we will use X_i^n to represent the coordinate of node i at "discrete time" n.

Without loss of generality, consider that a node (called the embedding node) computes its coordinate through a series of embedding steps, where each embedding step represents a coordinate adjustment based on a one-to-one interaction with another node, called a peer node (e.g. peer nodes are called neighbors in Vivaldi, and landmarks or reference points in NPS). Note that when the embedding protocol requires that a node uses several peer nodes simultaneously for repositioning, for the purpose of our modelization, we simply consider that each peer node corresponds to a distinct embedding step, each taking place at "successive" discrete times.

At every embedding step, the "fitness" (or "correctness") of the embedding node coordinate is assessed by computing the deviation between the measured RTT towards the corresponding peer node and the one estimated in the coordinate system. More precisely, suppose that at its n^{th} embedding step, embedding node i has current coordinate X_i^n and uses peer node j with current coordinate X_j^n. Suppose that the RTT between these nodes, measured during this embedding step, is RTT_{ij}^n. The fitness of the embedding node coordinate can then be computed as the *measured relative error* $D_n = \frac{|\|X_i^n - X_j^n\| - RTT_{ij}^n|}{RTT_{ij}^n}$. The goal of *any* embedding system, regardless of the embedding method proposed and/or the geometric space structure, is to minimize a "cost" indicator (e.g. mean square error) that captures the measured relative error that could be observed between any node and any other node in the system, at any time.

As the measured relative errors are fundamental performance indicators to all embedding systems, it seems natural to develop a model that captures their dynamic characteristics, although we note that relative errors often have complex behavior (and may thus not be a natural choice from a modeling perspective).

Measured relative errors are subject to fluctuations of the RTT for the reasons mentioned above, namely transient network congestion and operating system scheduling issues. To isolate the impact of these RTT fluctuations on anomaly detection, we introduce Δ_n, the *nominal relative error* that our node under consideration would have obtained at its n^{th} embedding step if the RTTs in the network had not fluctuated. An anomaly becomes simply a large deviation of measured relative error D_n from its nominal value defined by Δ_n.

Because many sources contribute to the deviation of D_n from its nominal value (RTT measurement error, RTT fluctuations, errors in node coordinate), it is reasonable to suppose that they relate to each other as follows,

$$D_n = \Delta_n + U_n \tag{4.1}$$

where U_n is a Gaussian random variable with mean zero and variance ν_U.

We now focus on the dynamics of the system in its nominal regime where RTTs do not fluctuate. In the absence of complete and accurate knowledge of the system, nodes keep on adapting the nominal relative error on a pairwise basis with their peer nodes, aiming to optimize the cost indicator. This adaptation is subject to an error caused by the other nodes in the system adapting their coordinate (and corresponding relative error) in a completely distributed way. We thus define the *system error* W_n which represents the impact of other nodes on the positioning of a node at embedding step n. Since the system error at a node results from many contributing sources, it is also reasonable to assume that it is a white gaussian process (with mean $\bar{w}$ and variance ν_W)[1].

Because of the nature of large-scale embedding processes, the nominal relative error Δ_n can be deemed to follow a stochastic process that converges to some stationary regime characterized by a positive average. As a first approximation, the process Δ_n could be modeled as a first order Auto Regressive (AR) model:

$$\Delta_{n+1} = \beta\Delta_n + W_n. \tag{4.2}$$

where β is a constant factor strictly less than one otherwise the relative error does not converge to a stationary regime independently of the initial condition. This equation captures the dynamic evolution of the nominal relative error of a node through successive embedding steps.

Equations 4.2 and 4.1 define a linear state space model for the relative error of a node. Our goal is to devise a way to obtain relative error predictions from this model. Because of the linear properties of the model, a Kalman filter can be used to track the evolution of the nominal relative error and obtain a *predicted relative error* $\widehat{\Delta}_{n|n-1}$ (see section 4.3.1).

[1] The value $\bar{w}$ accounts for the drift that has been observed in positioning systems [18].

The idea behind this strategy is that if the stochastic space model, and especially its associated Kalman filter, are calibrated within a clean embedding system, then a simple hypothesis test can be used to assess whether the deviation between the measured relative error and the predicted relative error, observed at a given embedding step, is normal or is the fruit of anomalous or malicious activity from the peer node. From this perspective, even if the state space model considered is crude, its quality should be evaluated based on the final outcome in terms of probability of detection and false positive rate. We will see in the evaluation section (section 4.6) that this model achieves very good performance.

4.3.1 Kalman Filter Equations

The Kalman filter is used here to estimate Δ_n given the set of previously measured relative errors $\mathcal{D}_0^n = \{D_0, \dots, D_n\}$. Under the hypothesis of a gaussian noise process in the underlying state space model, the Kalman filter gives the Least Mean Squared estimates of Δ_n, $\widehat{\Delta}_n$. Moreover, it gives the quality of these estimates through an evaluation of the mean squared error i.e., $E[(\widehat{\Delta}_n - \Delta_n)^2]$. This last value could be used to detect anomalies through large deviations of the measured relative error from its mean.

We will assume here that all the parameters of the space model given in Eq. (4.1) and Eq. (4.2) are known and given. In the next section we will describe how to derive these parameters.

Let us denote by $\widehat{\Delta}_{i|i-1}$ the estimation of Δ_i knowing the observations of network delay up to time $i-1$, and $\widehat{\Delta}_{i|i}$ the estimate after the measurement D_i is done. Similarly, let $P_{i|i-1}$ be the estimated *a posteriori* error variance at time i knowing the observations up to time $i-1$, and let $P_{i|i}$ be the estimation of the *a posteriori* error variance after D_i is known. The Kalman Filter is composed of two steps that are iterated. The first step is called the prediction step and the second one the update step.

In the prediction step, the value of $\widehat{\Delta}_{i|i-1}$ is calculated based on $\widehat{\Delta}_{i-1|i-1}$ as :

$$\widehat{\Delta}_{i|i-1} = \beta\widehat{\Delta}_{i-1|i-1} + \bar{w}.$$

The *a posteriori* error variance of this estimate is :

$$P_{i|i-1} = \beta^2 P_{i-1|i-1} + \nu_W.$$

In the update step, $\widehat{\Delta}_{i|i-1}$ is updated to integrate the observed measurement D_i :

$$\widehat{\Delta}_{i|i} = \widehat{\Delta}_{i|i-1} + K_i(D_i - \widehat{\Delta}_{i|i-1})$$

where K_i denotes the updated gain and is obtained as :

$$K_i = \frac{P_{i|i-1}}{P_{i|i-1} + \nu_U}.$$

The *a posteriori* error variance of this estimate is :

$$P_{i|i} = \frac{\nu_U}{P_{i|i-1} + \nu_U} P_{i|i-1}.$$

The value $\eta_i = D_i - \hat{\Delta}_{i|i-1}$ is called the innovation process and is the main process to observe for anomalous behavior detection (see section 4.5.1). The innovation process is a white (meaning that it is an independent process) gaussian process with a mean 0 and a variance equal to $\nu_{\eta,i} = \nu_U + P_{i|i-1}$. Abnormality simply amounts to a significant deviation from the nominal values of the innovation process characterized by the Kalman filter.

To run the Kalman estimation, we need as initial values the system state value w_0 and the *a priori* state variance $P_{0|0} = p_0$. These two values are estimated during the parameters calibration step.

4.3.2 Calibration of the Kalman filter

Before running the estimation using the Kalman filter, the values of the filter parameters $\theta = (\beta, \nu_W, \nu_U, w_0, p_0)$ have to be computed. For this purpose we need to calibrate these parameters over coordinate measurements collected during a stationary and cheater-free period. The calibration can be done using a maximum likelihood criteria (choosing parameter values such that the likelihood of observing the measurements is maximized) by applying the Expectation Maximization (EM) method. We follow the approach presented in [60] for the EM derivation.

In the following, we give a brief description of the maximum likelihood estimation criterion and the EM method, we are using in this section to calibrate our filter.

The maximum likelihood estimation criterion

Maximum likelihood estimation (MLE) is a statistical method used to make inferences about parameters of the underlying probability distribution from a given data set [61]. Basically, this method allows us to infer the parameters of a distribution given a sample of data $X = X_1, \ldots, X_n$. Commonly, one assumes the data are independent, identically distributed (i.i.d) drawn from a particular distribution with unknown parameters and uses the MLE technique to create estimators for these unknown parameters.

Let us consider a family of distributions P_θ indexed by a parameter (which could be a vector of parameters) θ that belongs to a set Θ. Let $f(X|\theta)$ be either a probability function (in case of discrete distribution) or a probability density function (continuous case) of the distribution P_θ. Given our i.i.d. sample $X_1, \dots, X_n$ with unknown distribution P_θ from this family, i.e. parameter θ is unknown. A likelihood function is defined by:

$$L(\theta|X) = f(X_1|\theta) \times \ \dots \ \times f(X_n|\theta).$$

If our distributions are discrete then the probability function $f(x|\theta) = P_\theta(X = x)$ is the probability to observe a point x. $L(\theta) = f(X_1|\theta) \times \ \dots \ \times f(X_n|\theta) = P_\theta(X_1) \times \ \dots \ P_\theta(X_n) = P_\theta(X_1, \dots, X_n)$ is the probability to observe the sample $X_1, \dots, X_n$ when the parameters of the distribution are equal to θ.

Suppose that there exists a parameter $\hat{\theta}$ that maximizes the likelihood function $L(\theta)$ on the set of possible parameters, i.e.

$$L(\hat{\theta}) = \max_{\theta \in \Theta} L(\theta)$$

Then $\hat{\theta}$ is called the Maximum Likelihood Estimator (MLE) of θ.

When finding the MLE, it is sometimes easier to maximize the log-likelihood function since

$$L(\theta) \to maximize \Leftrightarrow log(L(\theta)) \to maximize$$

maximizing $L(\theta)$ is equivalent maximizing $\log L(\theta)$. Log-Likelihood function can be written as $\log L(\theta) = \sum_{i=1}^{n}(\log f(X_i|\theta))$.

To summarize, one needs to recall that the MLE criterion chooses the parameter $\hat{\theta}$ that maximizes the probability of seeing the observed data given that their distribution follows these parameters. The log of the likelihood is often used instead of true likelihood because it leads to easier formulas, but still attains its maximum at the same point as the likelihood.

The Expectation maximization (EM) algorithm is a valuable approach for maximum likelihood parameter estimation. In the next paragraph, we use this method to calibrate our Kalman filter parameters, i.e . find the values for the filter parameters that maximize the probability of a sequence of observations.

Calibration by EM method

Let's assume that $\mathcal{D}_0^N$ is the set of all measured prediction errors, $\mathcal{D}_0^N = \{D_0, \dots, D_N\}$ and let $\Delta_0^N = \{\Delta_0, \dots, \Delta_N\}$ be the set of nominal relative errors.

As all the noise processes are assumed to be gaussian, $\mathcal{D}_0^N$ and Δ_0^N will jointly follow a gaussian distribution. The log-likelihood of $\mathcal{D}_0^N$ and Δ_0^N, based on equations 4.2 and 4.1, can therefore be written as follows:

$$\begin{aligned}
&\log \mathrm{Prob}\{\mathcal{D}_0^N, \Delta_0^N\} = -\sum_{i=0}^{N} \frac{(D_i - \Delta_i)^2}{2\nu_U} \\
&- \frac{N+1}{2} \log \nu_U - \sum_{i=1}^{N} \frac{(\Delta_i - \beta\Delta_{i-1} - \bar{w})^2}{2\nu_W} \\
&- \frac{N}{2} \log \nu_W - \frac{(\Delta_0 - w_0)^2}{2p_0} - \frac{1}{2} \log p_0 \\
&- (N+1) \log 2\pi.
\end{aligned}$$

In a Maximum likelihood setting, we wish to find the values for the parameters that will maximize the above log-likelihood assuming that the sequence $\mathcal{D}_0^N$ has been observed. However as the sequence of system state Δ_0^N has not been observed, this maximization is not tractable directly and we have to apply the Expectation Maximization method [62]. This method transforms the maximization of the above likelihood function with unobserved system state sequence Δ_0^N to an iteration of successive steps where the system state sequence is assumed to be known and the parameters can be obtained through maximization of the likelihood function.

Each iteration of the EM method consists therefore of two steps. In the first step, we compute the expectation (over all values of the sequence of states Δ_0^N) of the log-likelihood, given the observed values of D_n and assume that the parameter values are equal to $\theta^{(k)}$. In a second step, the parameters $\theta^{(k+1)}$ are chosen so as to maximize the previously obtained likelihood expectation (see Appendix B). Next we explain these two operations with some further details.

Let the superscript (k) indicate the value of any parameter at the k^{th} step of the EM algorithm. As explained before, in the EM method, we need to estimate the value of the unobserved system states to be able to calculate the overall likelihood to maximize. The variables $\hat{\delta}_i^{(k)}$ are in fact those estimates at iteration k and $\hat{\pi}_i^{(k)}$ and $\hat{\pi}_{i,i-1}^{(k)}$ are the estimation error variances of this sequence of states:

$$\hat{\delta}_i^{(k)} = E[\Delta_i | \mathcal{D}_0^N, \theta^{(k)}],\ \hat{\pi}_i^{(k)} = E[\Delta_i^2 | \mathcal{D}_0^N, \theta^{(k)}],$$

$$\hat{\pi}^{(k)}_{i,i-1} = E[\Delta_i \Delta_{i-1} | \mathcal{D}_0^N, \theta^{(k)}].$$

Expectation step

The expected value of log-likelihood knowing the set of measured values $\mathcal{D}_0^N$ and the parameter $\theta^{(k)}$ is given by:

$$\begin{aligned}
\bar{L}(\theta, \theta^{(k)}) &= E[\log \mathrm{Prob}\{\mathcal{D}_0^N, \Delta_0^N\} | \mathcal{D}_0^N, \theta^{(k)}] \\
&= -\sum_{i=0}^{N} \frac{D_i^2 - 2D_i \hat{\delta}_i^{(k)} + \hat{\pi}_i^{(k)}}{2\nu_u} \\
&- \frac{N+1}{2} \log \nu_u - \sum_{i=1}^{N} \frac{\hat{\pi}_i^{(k)} + \beta^2 \hat{\pi}_{i-1}^{(k)} + \bar{w}^2}{2\nu_w} \\
&+ \sum_{i=1}^{N} \frac{\beta \hat{\pi}_{i,i-1}^{(k)} + \hat{\delta}_i^{(k)} \bar{w} - \beta \hat{\delta}_{i-1}^{(k)} \bar{w}}{\nu_w} \\
&- \frac{N}{2} \log \nu_w - \frac{\pi_0 - 2\hat{\delta}_0^{(k)} w_0 + w_0^2}{2p_0} \\
&- \frac{1}{2} \log p_0 - (N+1) \log 2\pi.
\end{aligned}$$

By replacing θ by its value at the k^{th} step of the EM algorithm, we obtain $\hat{\delta}_i^{(k)}$, $\hat{\pi}_i^{(k)}$ and $\hat{\pi}_{i,i-1}^{(k)}$, which gives the expected log-likelihood at the $(k+1)^{th}$ step. Next, we describe how to compute these values.

Calculating the parameters $\hat{\delta}_i$, $\hat{\pi}_i$, $\hat{\pi}_{i,i-1}$

As explained in section 4.3.2, the sequence of system states Δ_0^N is not observable. However, we need to give an estimate of this sequence to be able to obtain the likelihood. The sequence $\hat{\delta}_i$, $i = 1 \ldots, N$, is the sequence of system state estimates and $\hat{\pi}_i$, and $\hat{\pi}_{i,i-1}$ is the error variance of these estimates assuming that the sequence $\mathcal{D}_0^N$ has been observed. We resort to the solution in [60] for the calculation of these estimates using the overall measurements set.

The value $\hat{\delta}_i$, $\hat{\pi}_i$, and $\hat{\pi}_{i,i-1}$ are estimated using a Kalman filter, assuming that the system parameters are set as in $\theta^{(k)}$. However, there is a subtle difference with Kalman filter case described in section 4.3.1; here the estimates $\hat{\delta}_i$, $\hat{\pi}_i$ and $\hat{\pi}_{i,i-1}$ do not depend only on observations up to time i, but on future observations up to time $N \geq i$. The solution to deal with this is to implement a forward-backward approach similar to Baum-Welch filter used for finite EM algorithm [65].

For each value of the parameter set $\theta^{(k)}$, we first do a forward step following the relations given in section 4.3.1. The application of this forward step results in the values $\widehat{\Delta}_{i|i},\ i = 1, \dots, N$, $P_{i|i},\ i = 1, \dots, N$ and $P_{i|i-1},\ i = 1, \dots, N$.

To add the future measurements in the Kalman filter, a backward recursion step is also added. This step consists of the following equations:

$$\begin{cases} \widehat{\delta}_{i-1} = \widehat{\Delta}_{i-1|i-1} + J_{i-1}(\widehat{\delta}_i - \beta\widehat{\Delta}_{i-1|i-1}) \\ \widehat{\pi}_{i-1} = P_{i-1|i-1} + J_{i-1}^2(\widehat{\pi}_i - P_{i|i-1}) \\ J_{i-1} = \beta\frac{P_{i-1|i-1}}{P_{i|i-1}} \end{cases}$$

These equations give recursively the values $\widehat{\delta}_i$ and $\widehat{\pi}_i$. It still remains to obtain $\widehat{\pi}_{i,i-1}$. This last value could be obtained using the relation :

$$\widehat{\pi}_{i,i-1} = \nu_{i,i-1}^N + \widehat{\delta}_i\widehat{\delta}_{i-1}$$

where $\nu_{i,i-1}^N$ can be obtained through the backward recursion

$$\nu_{i,i-1}^N = P_{i-1|i-1}J_{i-1} + J_i(\nu_{i+1,i}^N - \beta P_{i|i})J_{i-1},$$

that is initialized by setting

$$\nu_{N,N-1}^N = (1 - K_N)\beta P_{N-1|N-1}.$$

Maximization step

In this step, the parameter vector at step $(k+1)$ is chosen to maximize the expected log-likelihood. This is done by solving the equation

$$\frac{\partial\bar{L}(\theta, \theta^{(k)})}{\partial\theta} = 0.$$

This results in the following set of equations:

$$\begin{cases} w_0^{(k+1)} = \hat{\delta}_0^{(k)} \\ p_0^{(k+1)} = \hat{\pi}_0^{(k)} - (\hat{\delta}_0^{(k)})^2 \\ v_u^{(k+1)} = \frac{1}{N+1} \sum_{i=0}^{N} D_i^2 - 2D_i\hat{\delta}_i^{(k)} + \hat{\pi}_i^{(k)} \\ \bar{w}^{(k+1)} = \sum_{i=1}^{N} \hat{\delta}_i^{(k)} - \beta^{(k+1)} \sum_{i=1}^{N} \hat{\delta}_{i-1}^{(k)} \\ \beta^{(k+1)} = \frac{\sum_{i=1}^{N} \hat{\pi}_{i,i-1}^{(k)} - \bar{w}^{(k+1)} \sum_{i=1}^{N} \hat{\delta}_{i-1}^{(k)}}{\sum_{i=1}^{N} \hat{\pi}_{i-1}^{(k)}} \\ v_w^{(k+1)} = \frac{1}{N} \sum_{i=1}^{N} \hat{\pi}_i^{(k)} + (\beta^{(k+1)})^2 \hat{\pi}_{i-1}^{(k)} + (\bar{w}^{(k+1)})^2 \\ -2\beta^{(k+1)}\hat{\pi}_{i,i-1}^{(k)} - 2\hat{\delta}_i^{(k)}\bar{w}^{(k+1)} + 2\beta^{(k+1)}\hat{\delta}_{i-1}^{(k)}\bar{w}^{(k+1)}. \end{cases}$$

By solving this set of equations we can obtain the vector $\theta^{(k+1)}$, then we iterate with the expectation calculation as described above.

We note that the complexity of the approach lies in the linear state space modeling phase by EM algorithm that incurs a number of iterations over N dimensional vectors, which is well within the capability of modern computers. We will see later that this phase has to be run on a subset of nodes (the Surveyors). On the other hand, predicting relative errors using the Kalman filter (section 4.3.1), which occurs on every node, only implies a few simple scalar operations and is negligible in terms of required computing power.

Finally, because we expect each of the innovation observation η_n to be inside a confidence interval of $\pm 2\sqrt{v_{\eta,n}}$ (where $v_{\eta,n}$ is the variance of the innovation process at time n) with a probability higher than 95%, when a Kalman filter yields 10 consecutive innovation observations outside such confidence interval, the filter is re-calibrated by re-applying the calibration procedure described in this section. Re-calibration is likely to occur following a significant change in the corresponding node's coordinate, caused by changes in network conditions.

4.4 Validation

To validate our model, we conducted simulations and PlanetLab experiments for both Vivaldi [15] and NPS [14]. Again, we consider Vivaldi as a prominent representative of purely peer-to-peer-based (i.e. without infrastructure support) positioning systems, while NPS is typical of infrastructure-based systems, where a hierarchy of landmarks and reference points govern the positioning of nodes.

As the goal of this section is to assess the fitness of the proposed model to represent the normal behavior of the embedding processes, all results presented were acquired in a clean environment with no malicious node. While the goal of the simulation studies is to assess our results for large scale coordinate-based systems, the PlanetLab experiments aim to show their applicability in real-world conditions.

The PlanetLab experiments were conducted over a set of 280 PlanetLab nodes spread world-wide. In this thesis, we discuss a representative set of experimental results conducted over several days in December 2006.

The simulations were driven by a matrix of inter-host Internet RTTs (the "King" dataset) to model latencies based on real world measurements. This dataset contains the pair-wise RTTs between 1740 Internet DNS servers collected using the King method [57] and was used to generate a topology with 1740 overlay nodes.

In the case of Vivaldi, each node had 64 neighbors (i.e. was attached to 64 springs), 32 of which being chosen to be closer than 50 ms. The constant fraction C_c for the adaptive timestep is set to 0.25, as proposed in [15]. A 2-dimensional coordinate space augmented with a height vector was used (see chapter 2, section 2.4.2).

For NPS, we considered an 8-dimensional Euclidean space for the embedding. We used an NPS positioning hierarchy with 4 layers. The top layer had a set of 20 well separated permanent landmarks. Each subsequent layer then had 20% of nodes randomly chosen as reference points. The security mechanism already proposed in NPS, shown to be too primitive in the previous chapter, was turned on and its sensitivity constant C was set to 4, as advised by authors in [14] (see chapter 2, section 2.3.4).

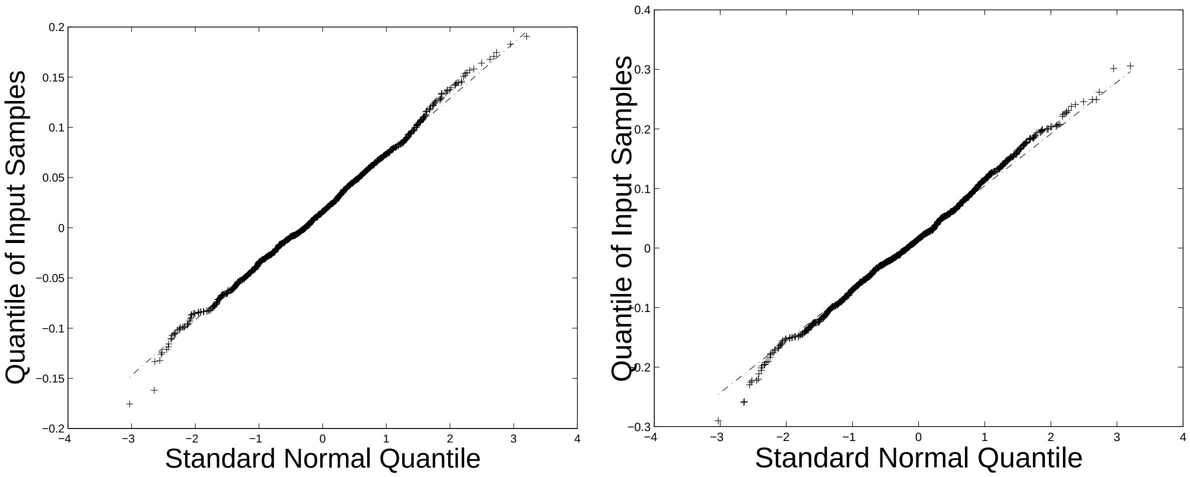
When needed, Surveyor nodes were chosen at random[2].

4.4.1 Assumption Validation

In section 4.3, the assumption that the system error W_n follows a gaussian distribution was made. This is fundamental to the applicability of the Kalman filter framework. For the purpose of validation, every node calibrated its own Kalman filter based on the observation of its own embedding, and we checked this assumption by applying the Lilliefors test [63], a robust version of the well known kolmogoroff-Smirnov goodness-of-fit test, to whitened filter inputs (see Appendix A for details). We observed that the Lilliefors test leads to only 14 gaussian fitting rejections in simulations (over 1720 samples) and 5 rejections in PlanetLab (over 260 samples). This test allows us to conclude

[2] Note that in NPS, all permanent landmarks also act as Surveyors.

that the hypothesis we took for the Kalman model is valid. In addition, we plot in figure 4.1 the Quantile-Quantile (QQ) plots of 2 innovation processes (for both Vivaldi and NPS) taken on PlanetLab nodes running their own Kalman filter. These plots, typical of observations on all nodes, show that each of these distributions indeed closely follows a Gaussian distribution.



(a) Vivaldi Case

(b) NPS case

Figure 4.1: Quantile-Quantile plot of 2 innovation processes.

4.4.2 Effective Behavior Representation

From section 4.3, it is clear that the Kalman filter model attempts to represent the behavior of the embedding process by capturing the dynamics of the system through its convergence behavior (by tracking of relative errors over time). In this section, we therefore assess the representational power of this approach by having each node calibrate its own Kalman filter from the measurements it observed during the embedding of its own coordinate, in a cheat-free regime. Then, once the model has converged at every node (i.e. the EM method has converged and the variations of all the θ components become smaller than 0.02), a new embedding process is started (i.e. the nodes forget their coordinates and rejoin the system). During this second embedding process, the prediction error, that is the absolute value between the error predicted by the node's Kalman filter and the measured actual error, is computed.

Figure 4.2 shows a typical evolution of actual (measured) relative errors and predicted errors for a node on PlanetLab (for Vivaldi, but similar behavior was observed

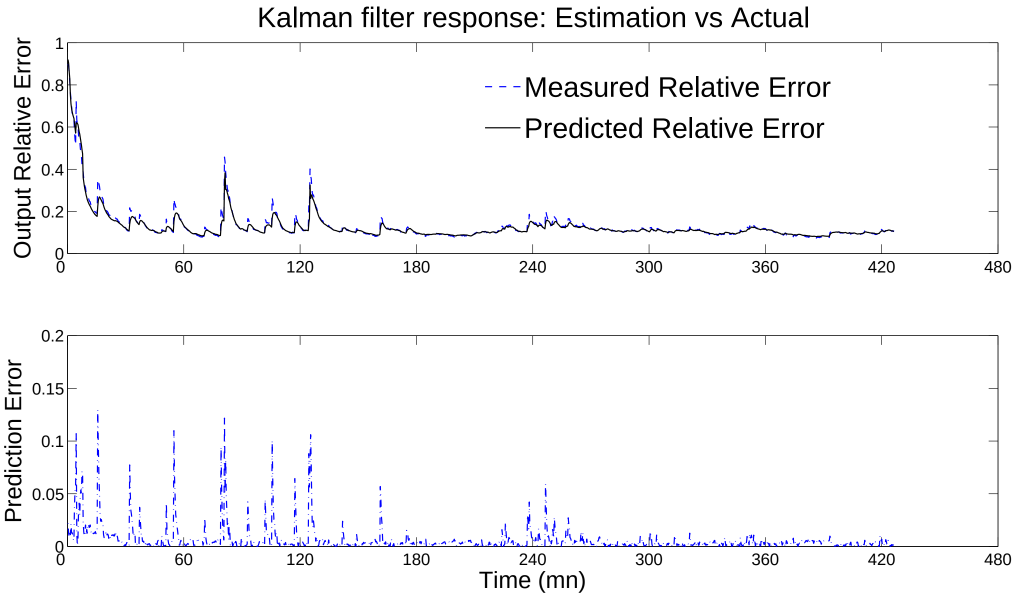


Figure 4.2: Prediction errors (PlanetLab node).

for NPS). The two curves of the top graph of the figure are so similar that they are almost indistinguishable. The bottom graph of the figure represents the prediction error which is the difference between these two curves (note the different scale used for this graph). We see that the prediction errors are small which shows that a node's calibrated Kalman filter can capture effectively the node's behavior "in the wild".

Figure 4.3 shows the cumulative distribution function (CDF) of all the prediction errors observed across all nodes in the system. This confirms that the vast majority of predictions are indeed excellent. This demonstrates the power and generality of the model in capturing the dynamics of the system and its adaptability to current system conditions.

However, there are a few "outlier" predictions with large errors. To better understand their nature, we show in table 4.1 the repartition of prediction errors in error intervals. The table shows the number of nodes with prediction errors in this interval, the number of occurrences of the smallest prediction error observed in the interval, and the number of occurrences of the largest prediction errors observed in the interval (e.g. Number of node(s)/number of observed min error/number of observed max error). We see that only a few nodes contribute (sometimes very many) large prediction errors. Looking further, we identified 3 nodes, all located in India, who contributed consistently to the "tail" of the CDF in figure 4.3. It is interesting to note that these

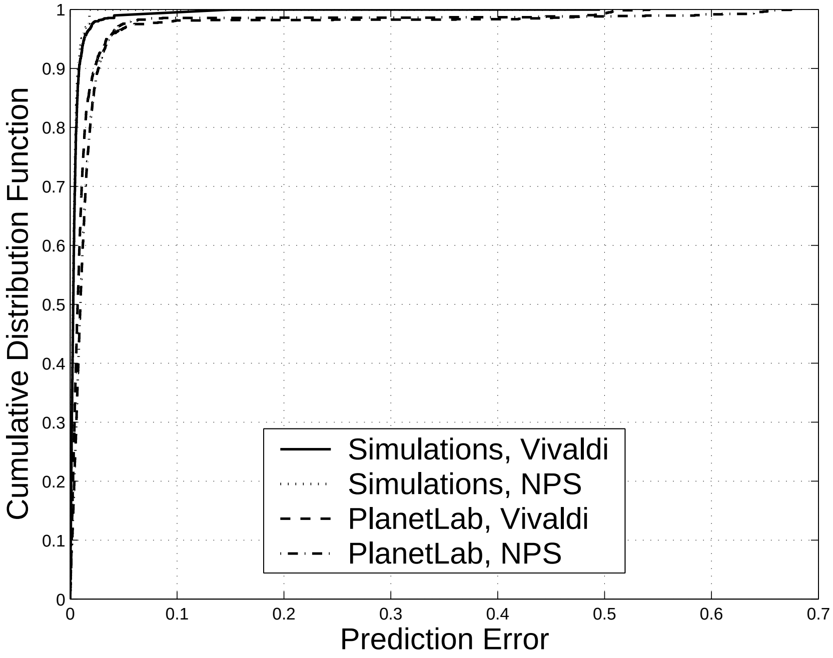


Figure 4.3: CDF of prediction errors.

nodes exhibited large (> 0.75) average measured relative errors during embedding, and were clearly having trouble with the embedding process itself, due to adverse network conditions.

4.4.3 Representativeness of Surveyors

If a subset of nodes in the system (called Surveyor nodes) are trusted and use each other exclusively to compute their own coordinates, they will be immune to the effects of malicious behavior during embedding. The premise of our work is that the “clean” (normal) system behavior thus learnt can then be shared with other nodes and used by these nodes to detect malicious behavior they may be subjected to by other nodes in the system. This obviously assumes that the behavior of the system as observed by Surveyors can approximate or represent well enough the normal behavior of the system as observed by other nodes in the absence of malicious behavior. In the following validation of this assumption, Surveyors are chosen at random in the node population.

Note that a random choice will give an upper bound on the number of Surveyors needed. Indeed, intuitively, Surveyors should be roughly uniformly distributed in the system to be representative of most other nodes. However, choosing nodes at random

Table 4.1: Prediction Error Histogram

Error Interval	(Vivaldi)	(NPS)
0.0-0.05	257/830/922	232/854/943
0.05-0.1	32/201/995	44/180/941
0.1-0.15	5/3/992	18/5/229
0.15-0.2	1/997/997	2/12/884
0.4-0.45	4/3/56	3/5/12
0.45-0.5	1/12/12	2/1/17
0.5-0.55	1/985/985	2/32/40
0.6-0.65	-	1/851/851

in the system does not yield a uniform distribution of Surveyors (i.e. “Surveyor clusters” appear due to the cluster structure of nodes themselves) and therefore not every new Surveyor increases representativeness. Consequently, more Surveyors must be chosen in order to achieve a good coverage of the system, than if they were placed more strategically. Nevertheless, the random choice method does provide general results, without the need to address the question of optimal Surveyor deployment strategy.

One of the first questions to answer is how many Surveyors are needed to be representative of the rest of the population. Noting that our model is based on measured relative errors and that each node in the system observes a series (i.e. distribution) of such errors, we characterize the system-wide relative error behavior as the CDF of the 95^{th} percentiles of the relative errors observed at each (normal) node (i.e. the distribution is made up of the 95^{th} percentile value observed at every node). We then compare this CDF with those of the 95^{th} percentiles of the relative errors observed across a varying population of Surveyors. The choice of the 95^{th} percentile is so that outliers, as observed in section 4.4.2, do not skew the results, while preserving a high degree of generality. Figure 4.4, obtained by simulations of Vivaldi, indicates that a population of Surveyors of about 8% of the overall population is closely representative of this overall population (because the CDF for these populations in the figure are similar).

To generalize this result, we then repeated the experiment, using both simulations

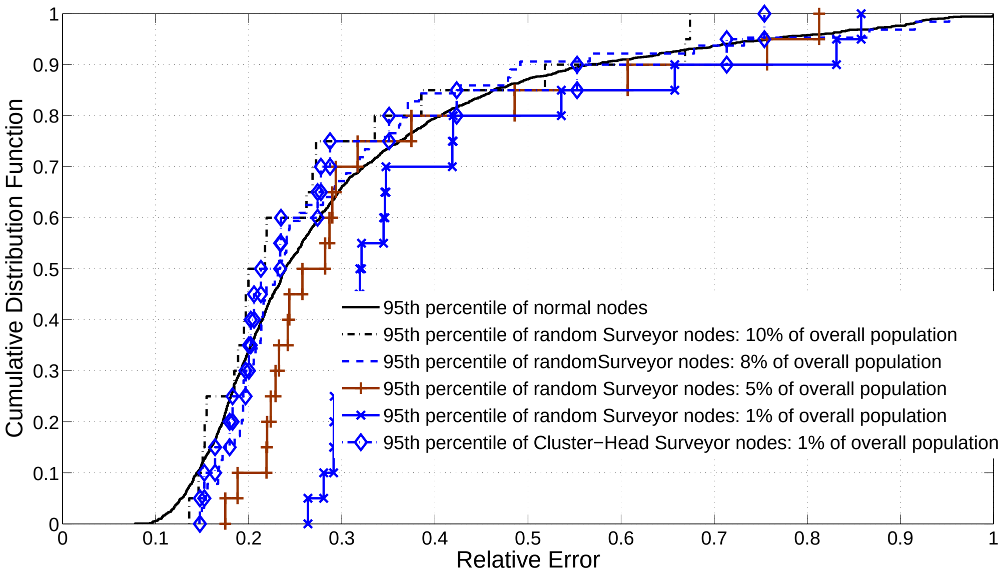


Figure 4.4: Impact of Surveyor population size on repretentativeness.

and PlanetLab measurements, on a Vivaldi system with 8% of Surveyors. We again chose to represent the system by the distribution of the 95^{th} percentile of the measured relative errors (figure 4.5).

These experiments confirm that less than 10% of randomly chosen Surveyor nodes is enough to gain a good representation of the system behavior. Similar results were observed for NPS. We note that 8% to 10% of the overall node population is a very stringent requirement for most practical purposes and can represent a huge number of nodes. However, as already pointed out above, random Surveyor deployment is not optimal and this value is an upper bound on the number of Surveyors needed.

To gain further insight into how conservative this upper bound may be, we tried a simple k-means clustering algorithm for Surveyor deployment. Figure 4.4 shows that when taking cluster heads as Surveyors, good representativeness can be achieved with roughly 1% of Surveyors. Although this does not give much indication as to what the lower bound on the number of Surveyors needed is in the case of optimal Surveyor deployment, it nevertheless shows that simple deployment methods can reduce requirements considerably and that the upper bound yielded by random deployment is indeed very conservative.

Having shown that a population of Surveyors can represent the overall system, the next question is how well the behavior of the system as captured by the Kalman

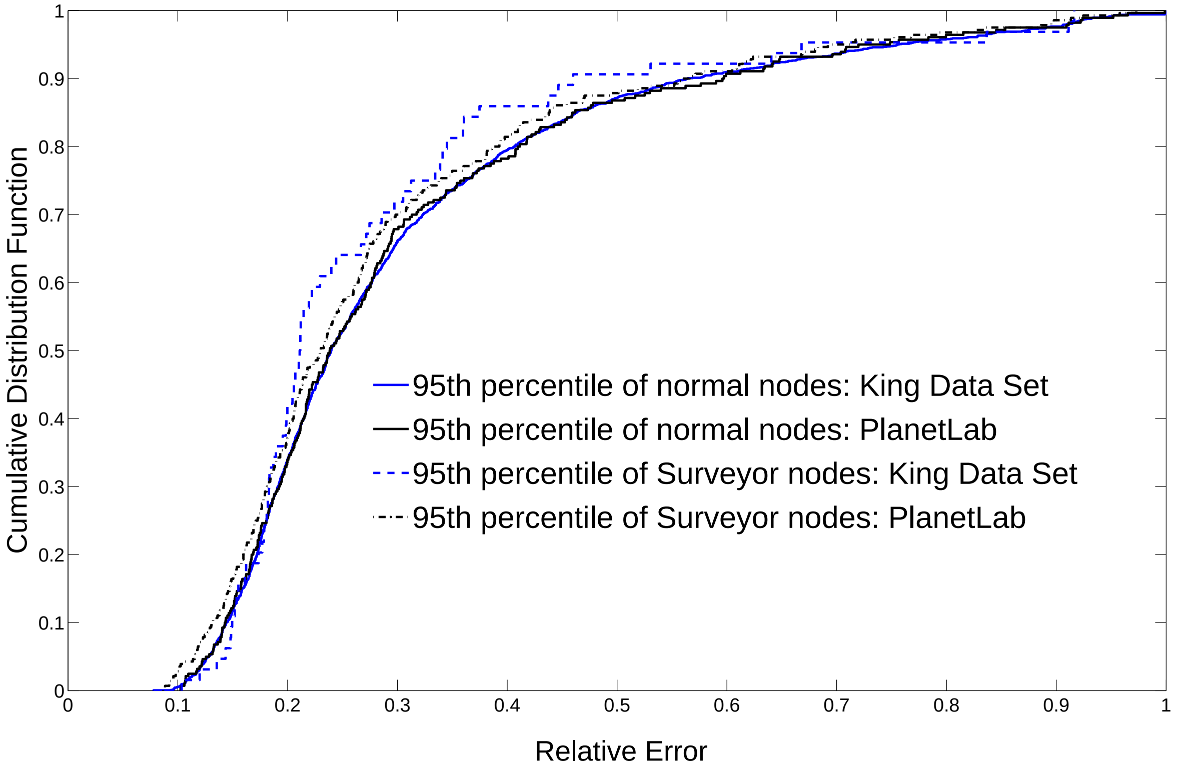


Figure 4.5: Representativeness with 8% Surveyor nodes.

filter calibrated by a Surveyor, can represent the behavior of a single (normal) node. To answer this question, we carried out an experiment where a population of nodes took part in a Vivaldi embedding on PlanetLab. Each node used the Kalman filter of every Surveyor and generated multiple prediction errors (one per Surveyor) at every embedding step.

Figure 4.6 shows the maximum prediction error yielded by each Surveyor, for each normal node in the system, observed during this experiment. What we observe is that although each normal node can find at least one Surveyor node whose Kalman filter yields very low prediction errors, not every Surveyor is a good representative for any given normal node. The Surveyor chosen as a representative by a normal node is therefore important to achieve good prediction performance (and thus good malicious behavior detection).

Figure 4.7 plots the prediction accuracy (measured as an average prediction error) against the distance (measured as an RTT) between a node and the corresponding Surveyor, as observed during the PlanetLab experiment. It is clear that better locality

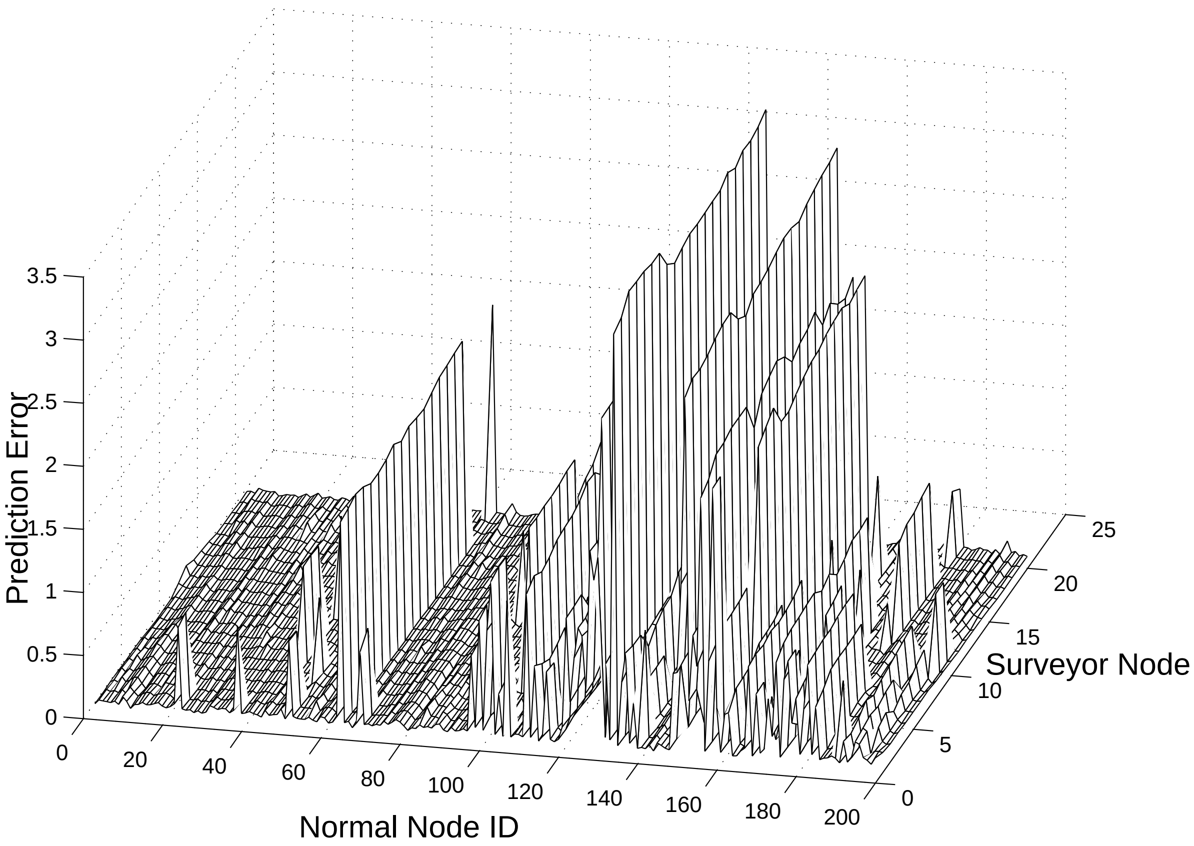


Figure 4.6: Maximum prediction errors with Surveyor filter parameters (PlanetLab).

between a node and its Surveyor yields more accurate predictions. This property seems intuitive, as a Surveyor closer in terms of RTT will also be closer in the geometric space, and will thus be more likely to experience dynamics of the coordinate system similar to that of the local area where the node resides. This is confirmed in figure 4.8 which shows the maximum prediction error, observed for Vivaldi on PlanetLab, when nodes use the closest Surveyor as their representative. Similar results were observed for PlanetLab experiments with NPS.

Finally, it is again important to note that all the results in this section were obtained with randomly deployed Surveyors. Strategically placing Surveyors to ensure a better coverage of the network and coordinate space, would simply improve the prediction accuracy, while reducing the number of Surveyors required.

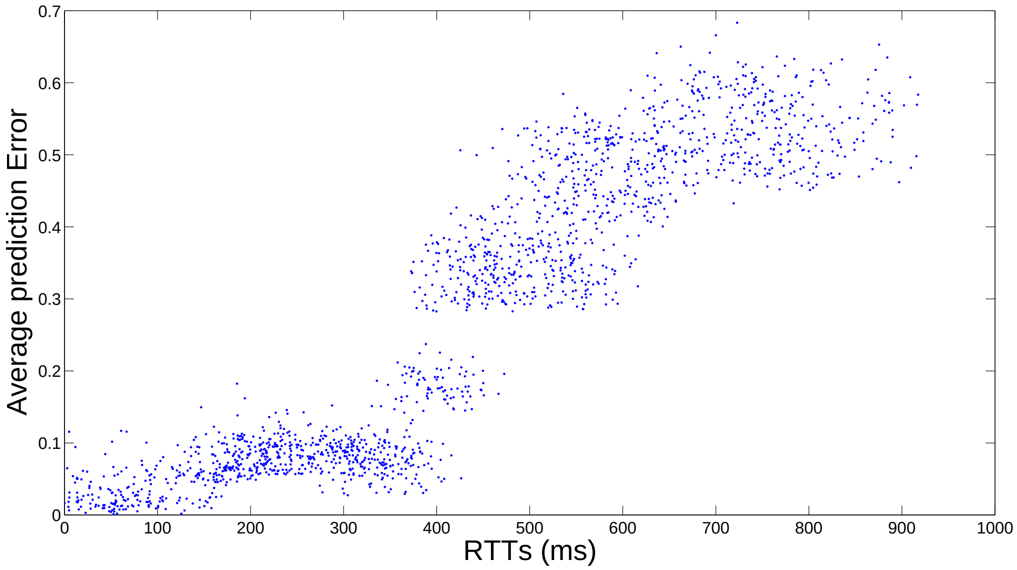


Figure 4.7: Correlation between 'Node-Surveyor' RTTs and estimation accuracy (PlanetLab).

4.5 Malicious Behavior Detection

The previous section has shown that normal node behavior can be modeled by a Kalman filter. More importantly, it has also been shown that this technique is powerful and robust enough that the normal behavior model captured on one node is readily and effectively applicable on other nearby nodes. This property leads to the idea of Surveyors.

Surveyors are a set of nodes in the coordinate space that exclusively use each other to compute their own coordinates. In other words, in Vivaldi, Surveyors only use other Surveyors as neighbors, while in NPS, they only use other Surveyors as reference points (note that in NPS, all landmarks also act as Surveyors, although not all reference points will be Surveyors). Of course, Surveyors can, and will be chosen as neighbors or reference points by other (non-Surveyor) nodes in the system, but the point is that a Surveyor adjusts its coordinate solely in response to embedding steps (i.e. measurements) with other Surveyors. If Surveyors run a clean version of the coordinate embedding software and they are carefully kept clean of malicious software, such as viruses or worms, that could implement malicious modifications to the embedding, then they can be considered as clean, honest nodes. Because Surveyors only interact with each other during their own embedding, they are therefore immune to malicious or anomalous

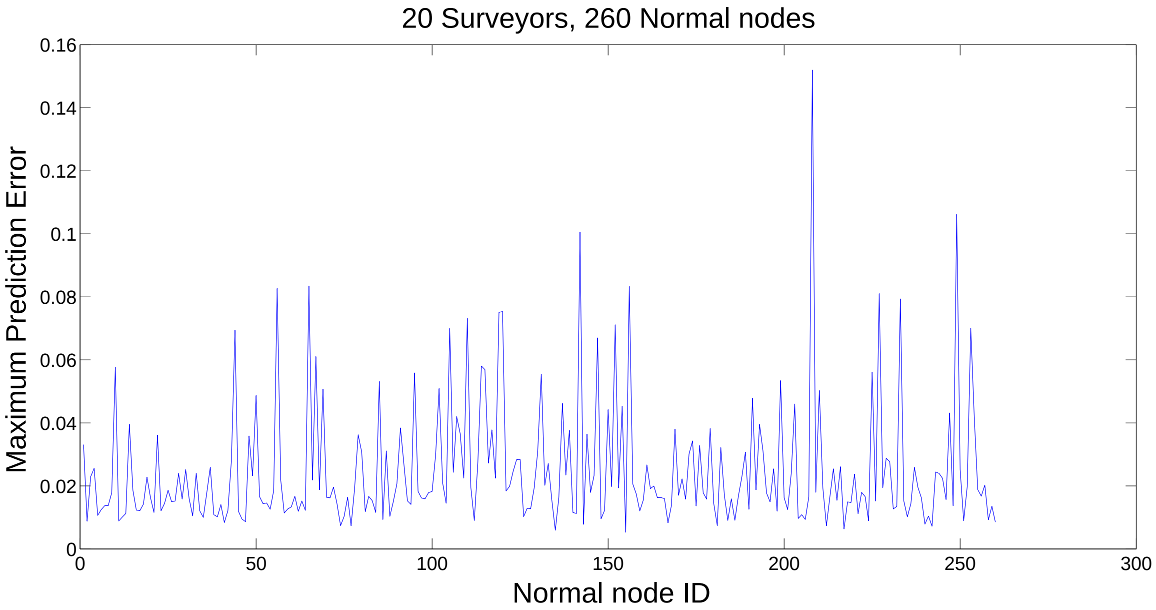


Figure 4.8: Maximum prediction errors with closest Surveyor.

behavior in the system, and they therefore observe the behavior of the system in clean, normal conditions. The idea is then to use the thus obtained normal behavior model as a basis for anomalous behavior detection at other nodes of the system. To do so, nodes use the parameters of the Kalman filter calibrated at a nearby Surveyor.

It is important to note that the proposed method is entirely distributed as each node has its own filter. Indeed, Surveyors calibrate and recalibrate their own filter as needed, depending on varying network conditions, and share the resulting filter parameters with other nodes, but they take no further active part in anomalous behavior detection at other nodes. When a node's filter needs re-calibrating (e.g. because it starts giving too many detection alarms), the node simply obtains fresh filter parameters from a Surveyor.

4.5.1 Anomalous Behavior Detection Method

At each embedding step, a node computes a measured relative error D_n towards a peer node. Recall from section 4.3 that the Kalman filter at the node can provide $\hat{\Delta}_{n|n-1}$, the predicted relative error from the previously measured relative errors. The innovation process of the Kalman filter yields the deviation between the measured and predicted relative errors, $\eta_n = (D_n - \hat{\Delta}_{n|n-1})$, which, in a system without malicious node, follows a zero-mean gaussian distribution with variance $\nu_{\eta,n} = \nu_U + P_{n|n-1}$ (also yielded by the filter).

This allows us to detect malicious behavior as a simple hypothesis test. Let H_0 be the hypothesis that the peer node has a normal behavior (i.e. it is honest). The hypothesis testing simply consists of assessing whether the deviation between the measured and predicted relative errors is normal enough under expected system behavior. Given a "significance level" α, which determines the "aggressivity" or "strictness" of the test, the problem is to find the threshold value t_n such that

$$P(|D_n - \hat{\Delta}_{n|n-1}| \geq t_n \mid H_0) = \alpha. \tag{4.3}$$

But since, under hypothesis H_0, $(D_n - \hat{\Delta}_{n|n-1})$ follows a zero-mean normal distribution with variance $\nu_{\eta,n}$, we also have that

$$P(|D_n - \hat{\Delta}_{n|n-1}| \geq t_n \mid H_0) = 2Q(t_n/\sqrt{\nu_{\eta,n}}), \tag{4.4}$$

where $Q(x) = 1 - \Phi(x)$, with $\Phi(x)$ being the CDF of a zero-mean, unit-variance normal distribution.

From equations 4.3 and 4.4, we therefore have

$$t_n = \sqrt{\nu_{\eta,n}} Q^{-1}(\alpha/2). \tag{4.5}$$

If the observed deviation exceeds the threshold given by equation 4.5, then the hypothesis is rejected, the peer node is flagged as suspicious, the embedding step is aborted and the measured relative error D_n is discarded (i.e. it is not used to update the state of the filter).

Note that a suspicious node, as detected by this test, is not necessarily associated with malicious intent, but could be caused by changing network conditions. Honest nodes classified as suspicious represent false positives and have little impact on the system as long as their occurrence is low. The trade-off between aggressivity and strictness of the test is represented by the so called ROC (Receiver Operation Characteristic) curves [64]. These curves plot the true positive rate versus the false positive rate, *i.e.* the probability of correctly detecting a malicious node versus the probability of labelling an honest node as malicious. In practice trying to increase the true positive rate (the probability of malicious node detection) comes at the cost of increasing the false positive rate.

4.5.2 Generic Detection protocol

In general, on identifying a peer node as suspicious, a node will replace it, that is choose a new neighbor in Vivaldi or a new reference point in NPS.

The only exception to this rule is when the node was embedding against the peer node for the very first time. In this particular case, the node uses its local error e_l[3] as an indicator of the confidence it has in its own coordinate, to carry out a second hypothesis test identical to that presented in the previous section, but this time with a confidence level of $e_l\alpha$. If the test is accepted, then the peer node gets a reprieve and is not replaced, so that a second embedding against this peer node will be attempted at a later time.

The main idea behind this potential reprieve for first-time peer nodes is that a node whose coordinate has already converged towards its true value can afford a few aborted embedding steps with very little impact on the accuracy of its coordinate. On the other hand, a new peer node which is in the process of joining the network may trigger the abortion of an embedding step, simply because its coordinate has not converged yet (as opposed to because it displays a malicious behavior). In this case, the reprieve simply gives time to the new (joining) peer node to converge before being identified as malicious. Of course an embedding node which is not confident in its coordinate must strive to reduce the number of aborted embedding steps so as not to compromise its convergence in the system, and will therefore grant fewer reprieves (because its e_l is higher) than a node that has already converged.

Finally, we use a simple mechanism for the selection of the Surveyor from which a node obtains its calibrated Kalman filter. All Surveyor nodes register with an infrastructure server (e.g. the membership server in NPS can act as Surveyor registrar, while in Vivaldi such server must either be introduced or at least integrated inside an existing bootstrap infrastructure). On joining the coordinate system, a node interrogates this server to obtain the identity of several (randomly chosen) Surveyors. The node then measures its distance to these Surveyors and selects the closest one as representative. From there on, the node fully complies to the embedding protocol rules, except that it will use our detection method to accept or reject embedding steps.

However, when the node has rejected half of its current peer nodes during a same embedding round, it will seek to acquire a new filter as the high rejection rate may

[3] e_l is the exponential moving weighted average of the measured relative errors of all previously completed embedding steps.

indicate that the filter parameters in use may have become stale (i.e. the filter needs "recalibrating"). The node then gets from its current Surveyor (or, as a fallback, any other Surveyors it knows, or the infrastructure server) the list of all the Surveyors it knows. After acquiring the current coordinates of these Surveyors, the node selects the closest one (in term of estimated distance) and obtains its Kalman filter parameters. Note that in the experiments we have carried out, which are described below, we observed very few "recalibrations", so this very simple Surveyor selection mechanism was appropriate. However, more sophisticated approaches can be considered if need be.

4.6 Evaluation

We evaluate the effectiveness of the simple anomalous/malicious behavior detection method in securing both Vivaldi and NPS. For each of these embedding protocols, we chose the most potent attack described in [2] and experimented with various populations of malicious nodes within the experimental set-up described in section 4.4. On PlanetLab, all these experiments were run concurrently so as to experience the same network conditions. In line with the results of section 4.4.3, the population of Surveyors was set to 8% of the overall population. Surveyors and malicious nodes were chosen at random.

4.6.1 Performance Metrics

To characterize the performance of our detection test, we use the classical false/true positives/negatives indicators. Specifically, a *negative* is a normal embedding step which should therefore be accepted by the test and completed. A *positive* is a malicious embedding step (i.e. where either, or both, the distance estimation and distance measurement between the node and its peer node have been tampered with) which should therefore be rejected by the test and aborted. The number of negatives (resp. positives) in the population comprising all the embedding steps is $\mathcal{P}_{\mathrm{N}}$ (resp. $\mathcal{P}_{\mathrm{P}}$).

A *false negative* is a malicious embedding step that has been wrongly classified by the test as negative, and has therefore been wrongly completed. A *false positive* is a normal embedding step that has been wrongly rejected by the test and therefore wrongly aborted. *True positives (resp. true negatives)* are positives (resp. negatives) that have been correctly reported by the test and therefore have been rightly aborted (resp. completed). The number of false negatives (resp. false positives, true negatives and true positives) reported by the test is $\mathcal{T}_{\text{FN}}$ (resp. $\mathcal{T}_{\text{FP}}$, $\mathcal{T}_{\text{TN}}$ and $\mathcal{T}_{\text{TP}}$).

We use the notion of *false negative rate* (FNR) which is the proportion of all the malicious embedding steps that have been wrongly reported as normal by the test, and $\text{FNR} = \mathcal{T}_{\text{FN}}/\mathcal{P}_{\text{P}}$. The *false positive rate* (FPR) is the proportion of all the normal embedding steps that have been wrongly reported as positive by the test, so $\text{FPR} = \mathcal{T}_{\text{FP}}/\mathcal{P}_{\text{N}}$. Similarly, the *true positive rate* (TPR) is the proportion of malicious embedding steps that have been rightly reported as malicious by the test, and we have $\text{TPR} = \mathcal{T}_{\text{TP}}/\mathcal{P}_{\text{P}}$.

The *true positive test fraction* (TPTF) is the proportion of positive tests that correctly identified malicious embedding steps ($\text{TPTF} = \mathcal{T}_{\text{TP}}/(\mathcal{T}_{\text{TP}} + \mathcal{T}_{\text{FP}})$).

4.6.2 Securing Vivaldi

We experimented with our detection scheme on a Vivaldi system subjected to a colluding isolation attack as described in the previous chapter and in [2]. In this scenario, malicious nodes are trying to isolate a target node, by repulsing all other nodes away from it. The malicious nodes agree on a large “exclusion” zone around the target node and randomly set their own coordinates outside this zone to try and attract honest nodes out of the exclusion zone. Note that an attacker always uses the same coordinate when lying to a given honest node.

Detection Method Performance

To evaluate the efficiency of the test, we first plot in figure 4.9, ROC (Receiver Operation Characteristics) curves observed for different significance levels (α) and several intensities of attacks. These plots show, for each significance level[4], the point corresponding to the false positive rate along the x-axis and to the true positive rate along the y-axis, with one curve per malicious group size (line x=y is plotted as a reference). Obviously, the closer to the upper left corner of the graph a curve is, the better, since such points correspond to high true positive rates (i.e. a high proportion of positives being reported as such by the test) for low false positive rates (i.e. a small proportion of negatives incorrectly reported as positives). We observe that from this perspective,

[4] Significance level values α always increase as a ROC curve is “followed” from the origin. In our experiments, we used values of 1%, 3%, 5% and 10% for α.

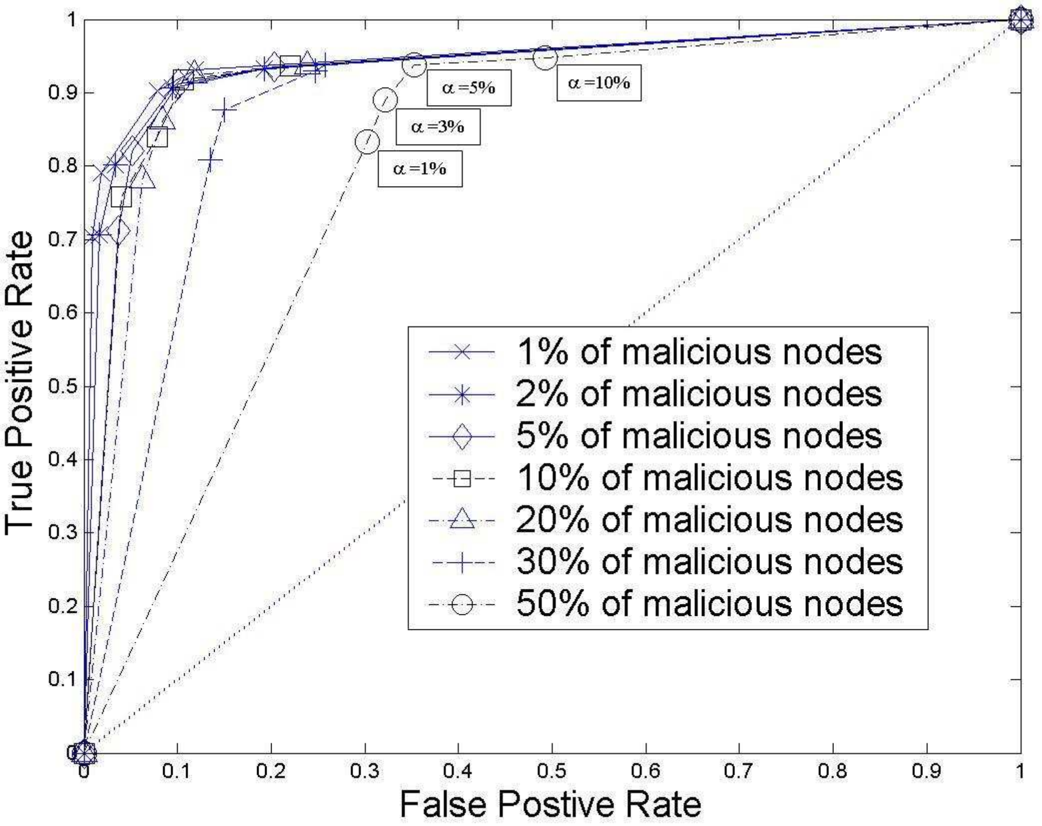


Figure 4.9: ROC curves. Each tick on the plots corresponds to a different value of the test's significance level (α).

the detection method can be considered to be excellent for 20% of malicious nodes or less, and still performs well even under heavy attack of up to about 30% of malicious nodes, while the power of the detection method naturally decreases as the malicious population becomes more significant. Another interesting properties of ROC curves is that they show the optimal range for the significance level. Indeed, as the slope of the ROC curve flattens, the increase in true positive rate is proportionally smaller than the corresponding increase in false positives. In other words, a higher significance level, although it always increases the true positive rate of the test, is not always productive as it eventually does more bad than good through increased false positive rates (i.e. the proportion of normal embedding steps that are aborted increases). This means that the significance level of the test should be set to a value that yields a point in the “elbow” of the ROC curve. Based on figure 4.9, we can deduce that a significance level of 5%

seems to be a good compromise.

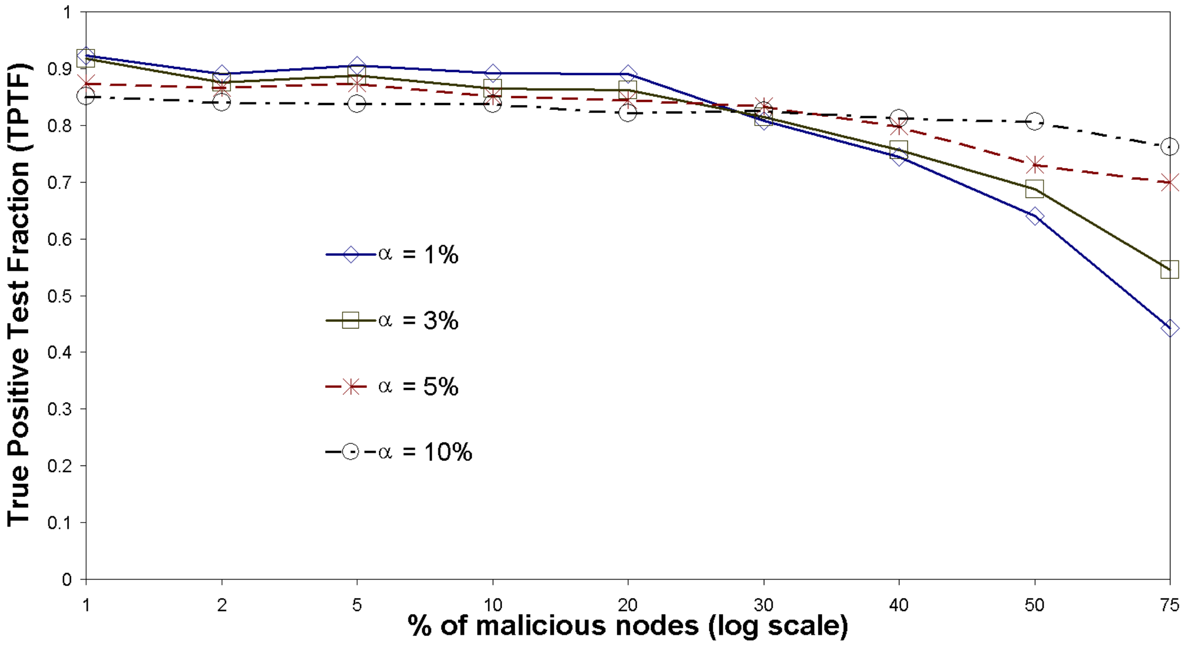


Figure 4.10: True positive test fraction.

Figure 4.10 shows the true positive test fraction of the detection method for various test significance levels under various intensity of attacks. We see that the proportion of positive tests that are true positives is constantly high, regardless of the significance level chosen, for moderate to quite significant proportions of malicious nodes in the population (up to 20% of malicious nodes). However, thereafter the proportion of correct positive tests starts to decrease, although the rate of decrease is inversely proportional to the significance level used. This is because a higher significance level produces more positive tests, catching most malicious embedding steps, and so many more false positives are needed to make up a significant proportion of these. In light of this, a significance level of 5% offers a good compromise.

Figures 4.11 and 4.12 show the false positive and negative rates respectively. As expected, a higher significance level results in a more aggressive test that incorrectly classifies a larger portion of normal embedding steps (figure 4.11) as malicious, while a more lenient test (lower significance level) wrongly reports a higher proportion of malicious embedding steps as normal (figure 4.12).

Incorrect test results do have a negative impact on the embedding system: false positives artificially reduce the size of available normal nodes that can be used for

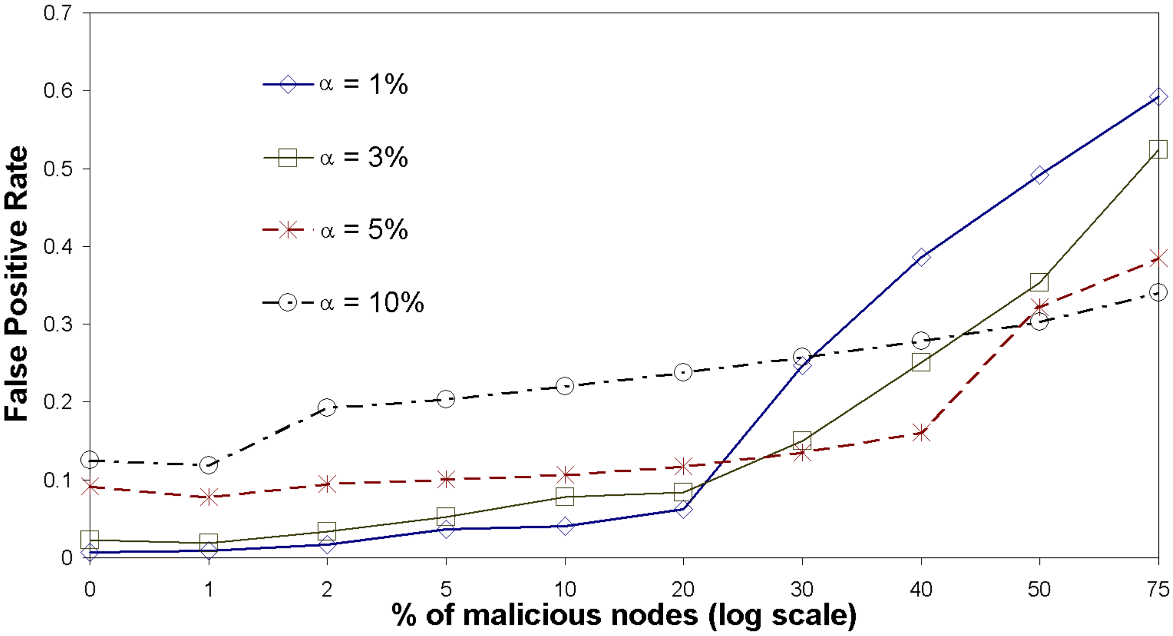


Figure 4.11: False Positive Rate.

normal embedding; false negatives give malicious nodes opportunities to corrupt and distort the coordinate space, which can propagate through the system and result in a greater proportion of normal nodes being identified as malicious (false positives) because of mis-positioning. This is exemplified in figure 4.11, where the false positive rate increases faster, as the population of malicious nodes increases, for lower values of the significance level of the test. Also, despite the fact that the false negative rate curves (figure 4.12) clearly exhibit negative slopes, one should note that these rates decrease much slower than the increase in malicious population. That is to say that as the number of malicious nodes in the system increases, the number of false negatives does increase, and more damage is incurred in the coordinate space. Although the accuracy of coordinate systems increases with the number of participating nodes, false negatives can therefore have a greater impact on the system than false positives and should therefore be thwarted in priority. As the false negative rates exhibited by tests with significance levels of 5% and 10% are roughly similar, while the more aggressive test yields proportionally a higher false positive rate, the significance level of 5% is a good compromise.

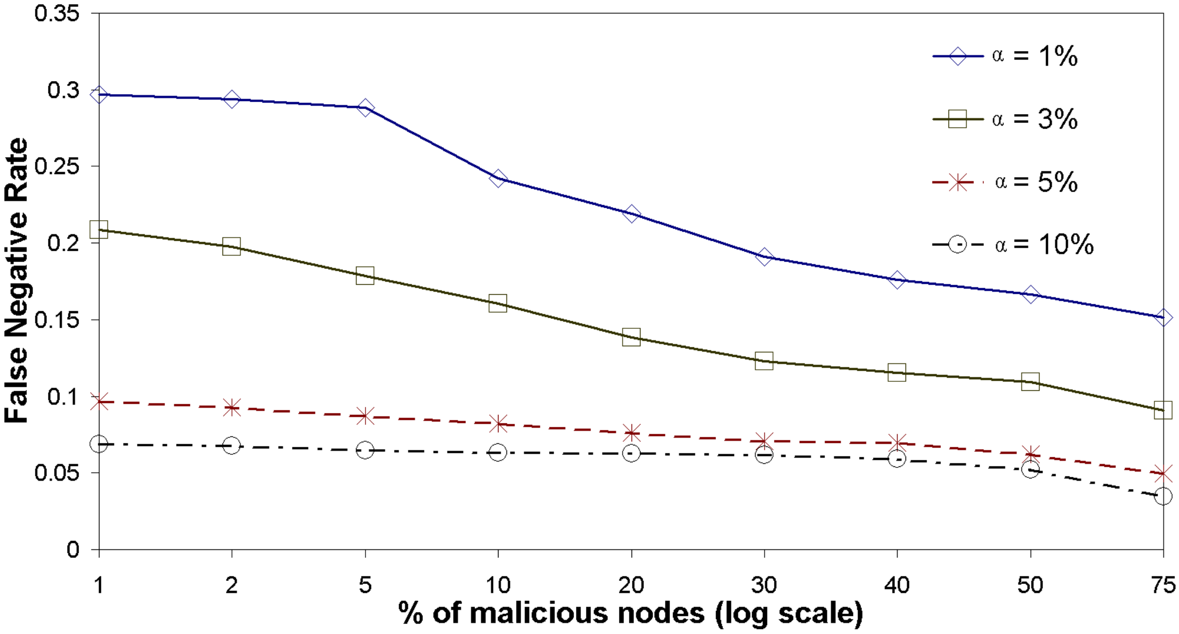


Figure 4.12: False negative rate.

Embedding System Performance

From section 4.6.2, it should be clear that a significance level of 5% gives the overall best test performance. We therefore set the significance level to this value and assess the resistance of a Vivaldi system under various intensity of attacks.

The cumulative distribution function of the measured relative errors, across all normal nodes, after convergence (in the sense of error convergence as defined in section 3) is shown in figure 4.13. We see that the detection mechanism renders the system practically immune to the attack, when the proportion of malicious nodes is 30%, or less, of the overall node population. Although the system does indeed show degraded performance for higher intensities of malicious attacks, the steeper slope of the CDF with detection, compared to the corresponding curve without (e.g. curves for 50% of malicious nodes), shows that the detection mechanism is not completely overwhelmed and still offers good protection by significantly reducing the impact of the attack.

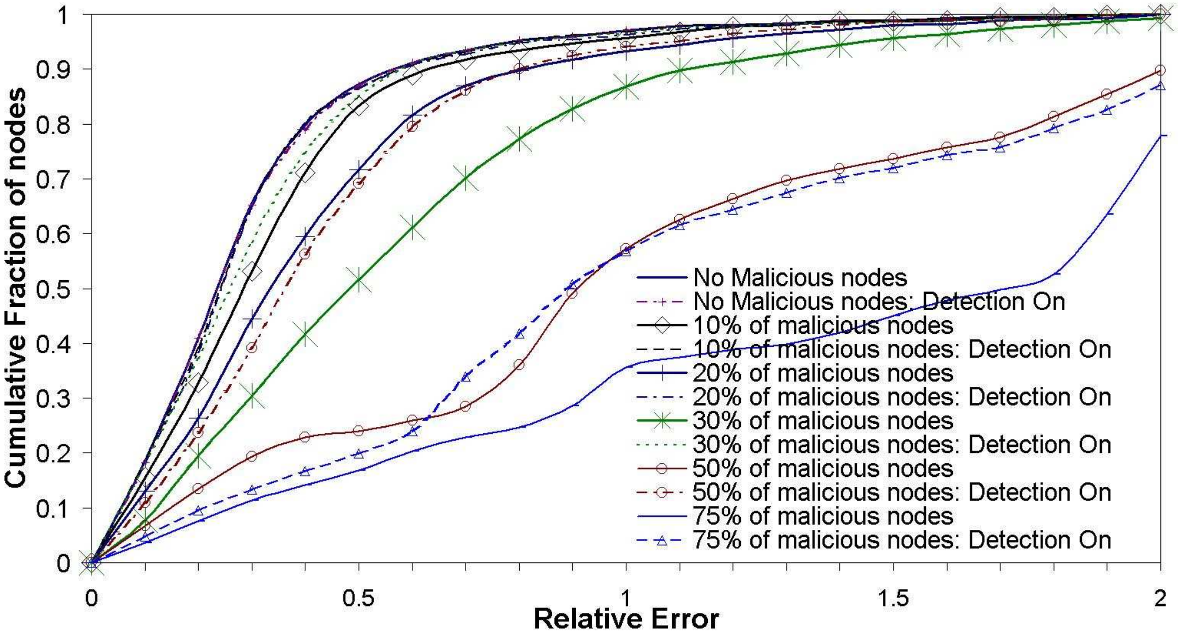


Figure 4.13: Distribution of measured relative errors.

4.6.3 Securing NPS

To test our proposed detection method in the context of the NPS coordinate system, we chose to study the effects of colluding isolation attack as described in the previous chapter and in [2]. The malicious nodes cooperate with each other and behave in a correct and honest way until enough of them become reference points at each layer. As soon as a minimum number of malicious reference points has been reached (in our experiments this number is set to 5) in a layer, these attackers identify a common set of victims (50% of the normal nodes they know from the layer directly below). When involved in the positioning of their victims, the malicious nodes agree to pretend they are all clustered into a remote (far away) part of the coordinate space and try and push the victims into a remote location at the ”opposite” of where the attackers pretend to be, in order to isolate the victims from the other nodes in the coordinate space. In order to evade detection, including the basic detection method proposed in NPS and which is *always* turned on in our experiments, the malicious nodes use the sophisticated anti-detection method proposed in [2] during their attacks.

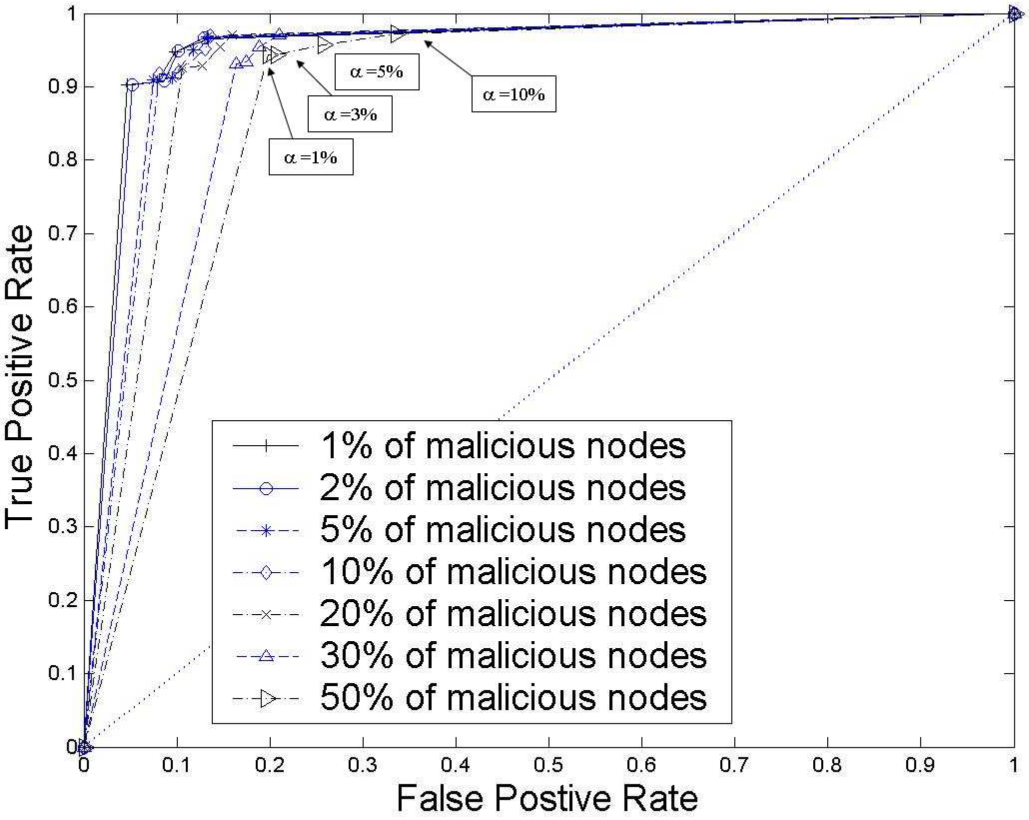


Figure 4.14: ROC curves.

Detection Method Performance

Figure 4.14 shows the ROC curves for the detection test in NPS. These curves show characteristics similar to those observed in the Vivaldi system (see section 4.6.2), albeit slightly better. In particular, these curves show that the detection method withstands heavier attacks better in NPS than in Vivaldi.

There are several reasons for this. First, the basic detection method in NPS works in concert with our own, providing greater opportunities to identify malicious behavior. Also, by its very nature, the embedding method in NPS is less prone to mis-positioning error propagation amongst normal nodes, as nodes in the lower layer do not take part in the embedding of other nodes. And finally, by design, the attack considered in this section makes fewer victims than that studied in section 4.6.2 (i.e. 50% of normal nodes as victims vs 100% in Vivaldi).

The same observation is also true for the false positive and false negative rates (not shown) with again, overall, a significance level of 5% seemingly offering the best compromise between "catching" malicious embedding steps while not being overly cautious and over-reacting to normal variations in network conditions.

The similarities between the test performance under NPS and Vivaldi, despite the different nature of the attacks under consideration and even differences in coordinate "structure" (two-dimensional with height for Vivaldi versus eight-dimensional for NPS), illustrates the generality of the proposed detection method. This is because our detection test is based on the modeling of a dimension-less quantity (the relative error) which is at the very core of *any* coordinates embedding system.

Embedding System Performance

We study the performance of the NPS embedding system when subject to increasing intensity of attacks, while being protected by our detection scheme. Note that in this section, "detection off" really means that our proposed detection mechanism is not used, but the basic NPS detection mechanism is still "on".

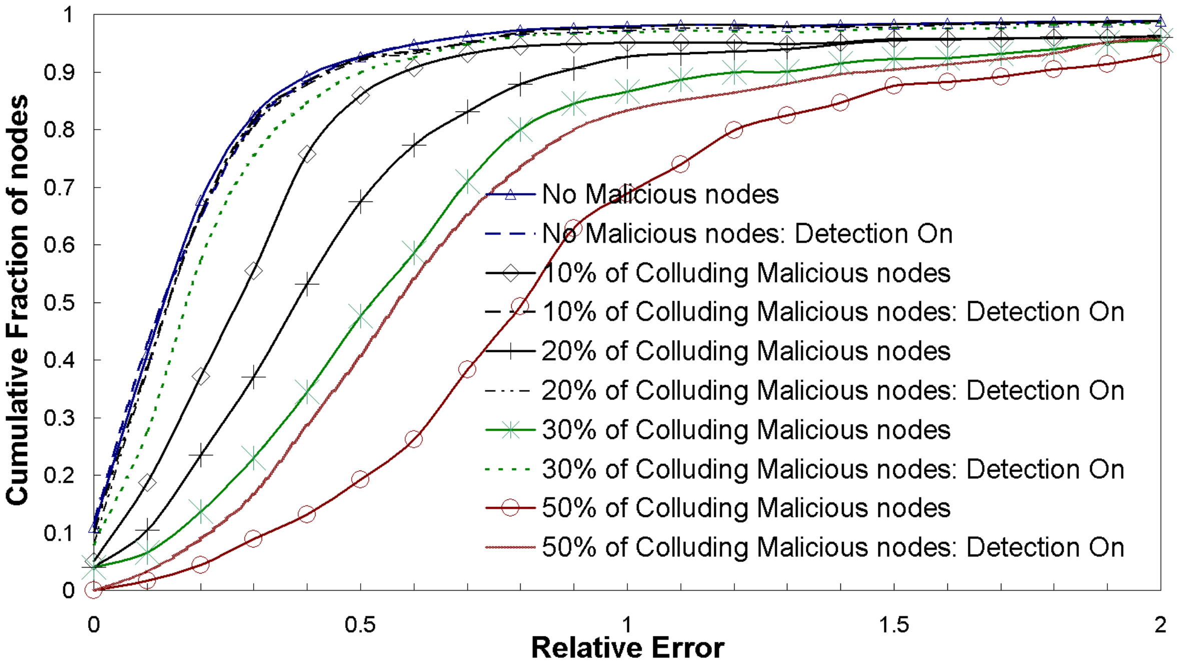


Figure 4.15: Distribution of measured relative errors.

Figure 4.15 shows the cumulative distribution function of relative errors in the system. We note again similarities with the dynamic behavior of similar Vivaldi systems, except that the tail of the CDF for 50% malicious nodes with detection is heavier than the corresponding curve in the Vivaldi case. Keeping in mind that in NPS not all nodes are victims and that not all normal nodes will propagate mis-positioning errors, this indicates that the attack is still quite effective against its victims, albeit "dampened" by the detection mechanism. This effect is compounded by the fact that, with our simple detection protocol, malicious nodes that have found their way into the layer hierarchy of NPS and act as Reference Points, do stay in place throughout the experiment, despite numerous detections of their corrupt embedding steps.

Nevertheless, the detection method proposed affords near immunity to the system up to rather severe attack conditions (e.g. about 30% of malicious nodes in the system).

4.7 Conclusions

We have presented a method for malicious behavior detection to secure the embedding phase of Internet coordinate systems. Our method does not rely on the geometric properties of the coordinate space, and is therefore unaffected by potential triangular inequality violations which often occur in the Internet [49, 48]. Instead, our detection test is based on the modeling of the dynamic relative errors observed in a clean system. The relative error is a dimension-less quantity which is at the very core of *any* embedding method, leading us to believe that our proposed detection test can effectively identify malicious behavior in very many embedding protocols and coordinate space structures that are under a potential very large range of attacks. The experiments presented in this chapter do show that the performance of the detection test is effectively the same in two different scenarios involving different embedding protocols and different attacks. As far as we know, this is the first such general detection test, capable of surviving sophisticated attacks. Also, we consider exclusively attacks aimed at distorting the coordinate space, carried out by nodes inside the embedding system. Our method thus succeeds where more obvious methods based on authentication would fail.

Nevertheless, with a trusted Surveyor infrastructure in place, it could be argued that using these Surveyors for positioning other nodes would ensure immunity to any insider attacks. For Vivaldi, using the Surveyors for positioning would mean that normal nodes only choose Surveyors as neighbors. The embedding performance of such

Vivaldi scenario is depicted by the "using dedicated Surveyors for embedding"-curve of figure 4.16, with the 1% Surveyors of the simple k-means deployment method. It can clearly be seen that such use of the Surveyor infrastructure trades embedding accuracy for increased security. A similar NPS scenario, where only Surveyors would be chosen as Landmarks and reference points, unsurprisingly led to embedding performance equivalent to a clean NPS system, as the hierarchical embedding structure in both systems are very similar. An NPS system where only Surveyors would be chosen as Landmarks and reference points actually looks like a hybrid system between GNP [11] and NPS: it is a fixed infrastructure system (like GNP), but with distributed Landmark coordinate computation and a hierarchical structure (like NPS). In both cases above, a clear scalability issue arises as the load on each Surveyor increases as their number decreases. In light of the discussion on strategic Surveyor deployment, as well as lower bound on the number of required Surveyors (see section 4.4.3), it is not clear that the solution of embedding against Surveyors only is practically viable. Even if it were, a hybrid solution, where Surveyors would be used for malicious activity detection under mild to medium attack intensities and exclusively used for embedding under more severe conditions, would be more accurate and afford better scalability.

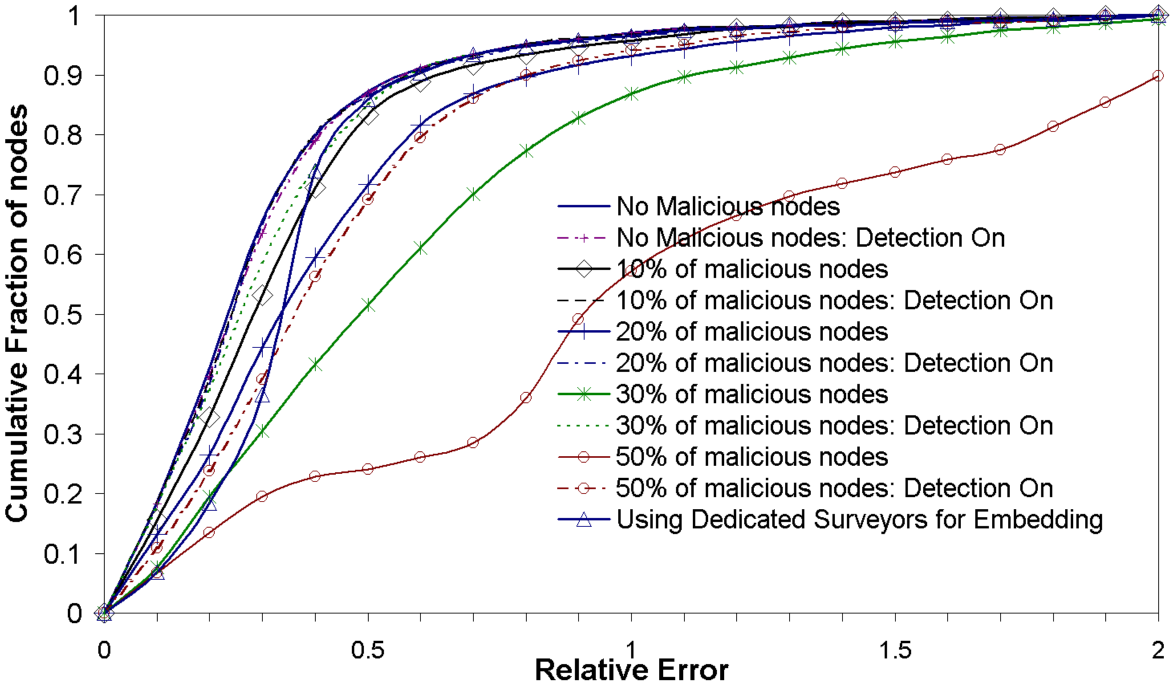


Figure 4.16: Distribution of measured relative errors.

Finally, even though the embedding phase of the system may have been secured, this would not prevent a malicious node from blatantly lying about its coordinate when a node requests them for simple distance estimation, during the “usage phase” of the coordinate service. This normal use of the coordinates service must therefore also be secured. We address this problem in the next chapter.

5

Securing the Distance estimation phase

5.1 Summary

In this chapter, we address the issue of asserting the accuracy of Internet coordinates advertised by nodes of Internet coordinate systems during distance estimations. Indeed, some nodes may even lie deliberately about their coordinates to mount various attacks against applications and overlays.

Our proposed method consists in two steps: 1) establish the correctness of a node's claimed coordinate by using the Surveyor infrastructure and malicious embedding neighbor detection proposed in our previous chapter on securing the coordinates embedding phase; and 2) issue a time limited validity certificate for each verified coordinate.

Validity periods are computed based on an analysis of coordinate inter-shift times observed by Surveyors. By doing this, each surveyor can estimate the time until the next shift and thus, can limit the validity of the certificate it issues to regular nodes for their calculated coordinates. Our method is illustrated on a trace collected from a Vivaldi system deployed on PlanetLab, where inter-shift times are shown to follow long-tail distribution (lognormal distribution in most cases, or Weibull distribution otherwise). We validate the effectiveness of our method by measuring the impact of a variety of attacks on distance estimates.

5.2 Introduction

The security of the embedding phase, or in other words the coordinate computation phase, of Internet coordinate systems has has been addressed in the previous chapter by proposing a general, embedding protocol-independent cheat detection mechanism based on the Surveyor infrastructure. However, ultimately, Internet coordinate systems are used to estimate distances between nodes, based on their coordinates only, even and all the more so if these nodes have never exchanged a distance measurement probe. Whatever mechanism is used to obtain a node's coordinate (e.g. direct exchange, DNS-like repository, etc.), each node must somehow report its own coordinate computed during the embedding phase of the system. This is because, for scalability reasons, nodes compute their own coordinates in a distributed way. This, of course, provides a malicious node with an opportunity to strike: in order to achieve some application-dependent goal or advantage (e.g. free-riding, denial-of-service, isolation, ubiquity gift, etc), a node can repeatedly lie about its coordinate. Simply lying about its coordinate could seriously disrupt the operations of Internet applications relying on coordinate-based systems for distance estimation. Several studies have quantified the impact of cheating on topology-aware Internet applications, and have shown that simple attack strategies can prove to be very effective [68, 69, 70]. This chapter addresses the question of guaranteeing the veracity of the coordinates advertised by nodes.

To do so, we propose to leverage the Surveyor infrastructure and embedding cheat detection test. More precisely, we propose that several Surveyors measure their distance to a node in order to verify the correctness of its claimed coordinate (using the cheat detection test). If all Surveyors agree that this coordinate is the node's true coordinate, a time limited validity certificate, including the certified coordinate, is issued to the node.

A certificate validity time period is necessary because, due to dynamic network conditions, nodes' coordinates vary in time. Upon a coordinate change, an honest node would stop using its current certificate and seek a certification of its new coordinate. On the other hand, a malicious node could keep using a certificate related to a previous position, hence a careful balance between scalability and certificate validity is desirable. To achieve this, one of our contributions is to study the coordinate inter-shift time (i.e. the time between coordinate changes at a node) as observed for a Vivaldi system running on PlanetLab. We found that the coordinate inter-shift times at most nodes follow a lognormal distribution, with the rare cases when this distribution is inappropriate being accounted for by a Weibull distribution (note these are both long-tail distributions).

In section 5.3, we study and characterize the coordinate inter-shift times and show that these times observed at Surveyors can adequately and statistically model inter-shift times at closeby nodes. Section 5.4 describes the certification procedure in detail, while performance evaluation of our proposal in the context of various attacks is presented in section 5.5.

5.3 Coordinates Evolution model

Our goal is to characterize and ascertain forward validity of nodes' coordinates during the distance estimation phase. We therefore concentrate on tracking the evolution in time of coordinates along the different axis of the coordinate space by observing their evolution in a clean system (without any malicious activity).

5.3.1 Experimental Set-up

A first step in our study consists in characterizing the dynamics of coordinates, as observed by nodes in a clean coordinates system, with no malicious node. For this first purpose, we used an initial trace of measurements obtained by running the Vivaldi System on 600 planetLab machines through a period of 18-days. To evaluate our method and demonstrate the validity and effectiveness of our proposal to deal with various attacks, we also used in a second step live-PlanetLab experiments, deployed as a Vivaldi service over a three-weeks period, .

In our initial trace, we kept the coordinates of 450 nodes, as several nodes, that have been down for more than one week or experienced connectivity troubles, have been filtered out. Some of them even do not respond to different RPC requests sent by the Vivaldi service.

In this trace, Vivaldi is using a 3-dimensional Euclidean space for the embedding. Each 10 minutes, corresponding to an embedding step, nodes are adjusting their coordinates based on a one-to-one interaction with another node, called a peer node. Finally, Vivaldi nodes are filtering the stream of latency measurements from a remote node and turn these into expected latency values, based on a Moving Percentile (MP) filter, a variant on the Moving Median filter, used to filter out heavy-tailed errors [18, 71]. This filter acts as a low-pass filter, where anomalies are ignored while a baseline signal passes through. This allows Vivaldi to smoothly adapt to shifts in the baseline (that BGP route changes cause for example). The goal of the latency filter is to summarize the measurements, providing a current and stable description of the expected latency between any two nodes. Ledlie et al. [18] found that using this simple latency filter with the a size of the history window equal to four observations and a percentile of 25% was a good predictor for the next observed latency (i.e. consider the minimum value of a four-latency measurements stream, for coordinates computation).

The PlanetLab experiments were conducted over a set of 280 PlanetLab nodes spread world-wide, running Vivaldi, as a coordinate-embedding service. For the purpose of our experimentations, we slightly modified the logging functions of the Vivaldi protocol. Each node is running several instantiations to allow us experimenting different parameters in similar conditions. Nodes are then updating their coordinates as needed, depending on the embedding step defined in each instantiation of the Vivaldi protocol. In the same way, the dimension of coordinates varies from one instantiation to another on the same node [1], and a node is acting as a malicious node or as a honest 'normal' node.

Each node had 20 neighbors (i.e. was attached to 20 springs), 10 of which being chosen to be closer than 50 ms. Again, the constant fraction C_c for the adaptive timestep is set to 0.25 (see chapter 2, section 2.4.2). When needed, Surveyor nodes were chosen that represent 8% of the overall population as discussed in the previous chapter.

5.3.2 Observations

Figure 5.1 shows a typical evolution of the coordinates of a Vivaldi node. Each sub-figure depicts the evolution of one coordinate component along the three axis.

We observe that the system after roughly 250 embedding steps, reaches a stationary regime, but coordinates continue to fluctuate. Looking further in the entire set of nodes' coordinates, we observe that coordinates steadily variate, moving away from the origin of the coordinate system. The rate of these variations is low enough to

[1] We experimented with different Euclidean embedding spaces and a 2-dimensional coordinate space augmented with a height vector.

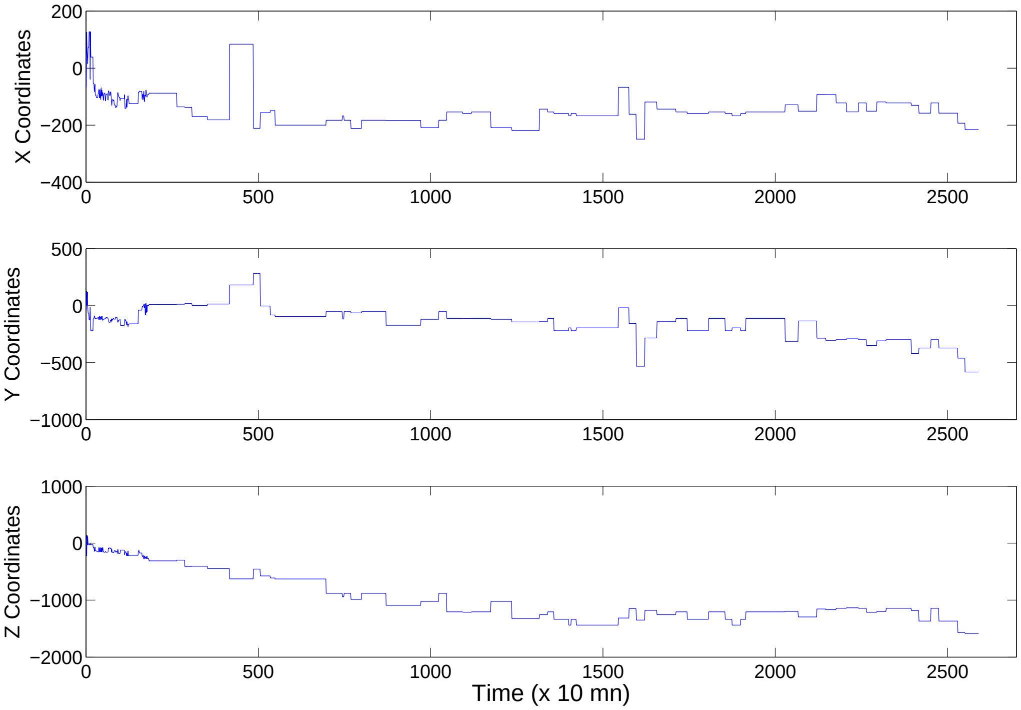


Figure 5.1: Typical Variations of a node's coordinate in the Vivaldi System.

allow distance estimation at the application level, but this prevents us from certifying permanent coordinates.

More specifically, because at any instant in time, the RTT that can be measured between two nodes depends on the state of the network (e.g. traffic load, state of queues in routers, etc), the exact value of the RTT varies continuously. However, it has been shown that RTT values in the Internet exhibit some stability in a statistical sense [59], with the statistical properties of RTTs exhibiting no significant change at timescales of several minutes. It is that property that embedding systems exploit to provide good distance estimates while only needing to have nodes adjust (recalculate) their coordinates on a periodic basis. Consequently, the node's coordinate can be viewed as a discrete stochastic process, computed at each embedding step.

Regardless of the dimensionality used by the coordinate-systems, our main goal is to assign to any coordinate given by a node, a reliability value that is telling the likelihood that this coordinate is still valid and has not changed. For this purpose we observe for our set of 450 planetlab nodes the inter-shift time distribution, corresponding to the amount of time (in terms of embedding steps intervals) during which, nodes stick to their positions, i.e. the coordinates do not change. This distribution is denoted T_i for each node i. It is important to note that although we observed that in our traces, a variation of one coordinate component was synonym to the variation of both others, we consider the inter-shift time as the laps of time corresponding to the non variation of all the coordinate components of this node.

Basically, we would like to determine which probability distribution is suitable to describe the inter-shift times. In the following, we will use our empirical data sets of inter-shift times to find the probability distribution that best describes the distribution values of each T_i.

5.3.3 Inter-shift Time Distribution Fitting

For choosing the best suitable distribution, we use a set of candidate distributions containing lognormal, Weibull, Rayleigh and Gamma distributions [2].

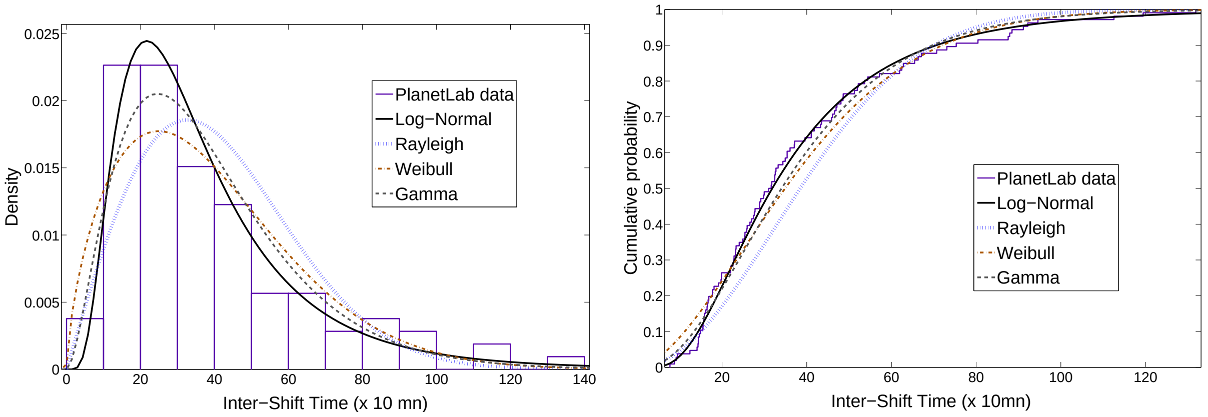
For each one of the 450 nodes in the dataset, we apply a two-step procedure. In the first step, we derive the maximum likelihood estimates of parameters of each distribution in the candidate set. The likelihood is the probability to get the observed inter-shift times for some hypothetic distribution. The estimates of the distribution parameters are then the values that maximize this likelihood.

In a second step, we used goodness of fit tests to evaluate if the hypothesis that the observed values T_i come from the candidate distribution can be rejected or not. The goodness of fit evaluation was done using the popular and robust Kolmogorov-Smirnov (K-S) test [72], applied at a significance level of 5% (see Appendix A for details).

We show in figure 5.2, the results of the fitting done on one of the datasets relative to node 181. Simple inspection by eye shows that the lognormal distribution produces the best fits among the set of distributions. This is validated by the K-S test that rejects all the candidate distributions but the lognormal.

Using the fitting procedure described above, we tried to fit the inter-shift datasets. The first interesting result we found is that all of the empirical distributions examined can be fitted to a known distribution. A large majority of distributions can be fitted into a lognormal distribution. The lognormal hypothesis was rejected for only 5 datasets out of the 450. Looking further in these 5 inter-shift datasets, we observed that they have

[2] The Gaussian distribution was not tested because the empirical distribution was not symmetrical around a mean.



(a) Histogram of the data and PDF of fitted distributions

(b) Empirical CDF and fitted CDFs

Figure 5.2: Density plots and CDF of a typical inter-shift distribution (Node 181).

Table 5.1: Results using the Kologorov Smirnov Test for fitting the inter-shift Times data.

Fitted Distributions	% of samples that passed the test
log-Normal	445/450
Rayleigh	2/450
Weibull	5/450
Gamma	6/450

a good fit with the Weibull distribution. Table 5.1 gives a summary of our findings for the Kolmogorov Smirnov goodness of fit tests.

5.3.4 Correlation between Surveyors and Nodes inter-shift Distributions

Having shown that inter-shift time distribution of our population of nodes in the Vivaldi system can be fitted in either a lognormal distribution in most cases or a Weibull distribution otherwise whose parameters can be obtained by a maximum likelihood method, the next question is how well the shifts as observed by Surveyor nodes can be used as representative of the shifts of regular nodes. Basically, if the inter-shift distribution of a node as seen by the surveyors is the same as the real inter-shift distribution, the former may be used as a reference to validate the node's coordinate.

The verification of this hypothesis is done by comparing the sequence of inter-shift as seen by surveyors and the real inter-shift of a node and asking if the hypothesis that these two sequences come from the same distribution can be rejected or not. The latter test is done using a two-sample Kolmogorov Smirnov Test (with a significance level of 5%) that precisely gives an answer to the previous question. For each of the 35 surveyor nodes, we applied therefore 415 two-sample K-S tests.

Then, we analyzed for each surveyor node the likelihood that the two-samples KS-test is rejected as a function of the distance (measured as an RTT) delay between the surveyor node and the tested regular node. The likelihood is obtained as the ratio between the number of nodes with a given delay that reject the test and the overall number of regular nodes. Figure 5.3 shows this rejection ratio versus the distance (measured as an RTT) between a node and the corresponding Surveyor, as observed in our traces.

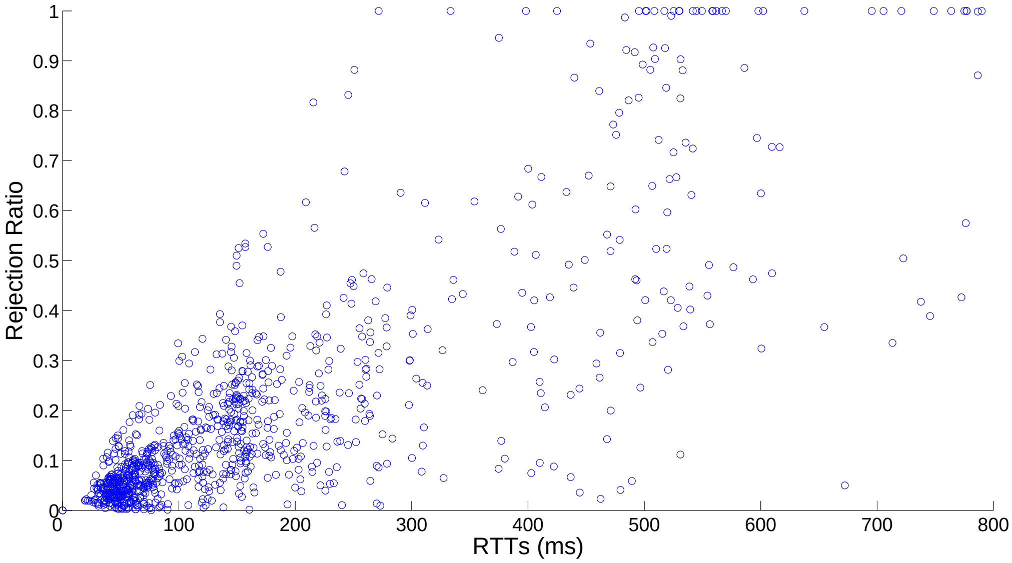


Figure 5.3: Correlation between 'Nodes-Surveyors' RTTs and the rejection Ratio.

Intuitively, a Surveyor should have tendency to have the same inter-shift distribution as nodes that are close by in terms of RTT as they are more likely to experience similar dynamics of the coordinate system. Figure 5.3 validates this intuition and shows that better locality between a node and its Surveyor yields more accurate fittings.

We therefore apply the following heuristics: the inter-shift time distribution of the closest Surveyor is used as the representative distribution for regular nodes.

5.4 Coordinate Certification

The method we propose to certify Internet coordinates consists in two steps:

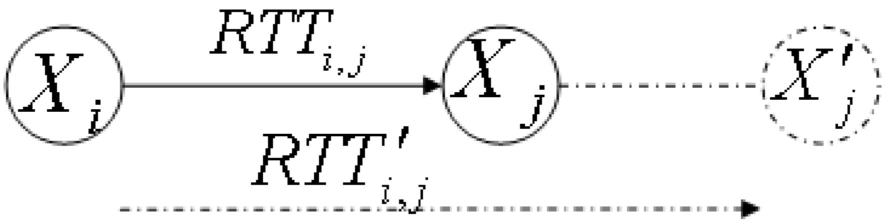
1. the node coordinate verification test
2. computation of an estimated validity period for this coordinate.

The coordinate verification test leverages the Surveyor infrastructure and malicious embedding neighbor detection, while the validity period estimation is exploiting the inter-shift distributions we have studied in the previous section.

5.4.1 Coordinate Verification

Principle

Recall from the previous chapter that our method for malicious embedding neighbor detection is based on a model of the evolution of a nodes's observed relative errors, during embedding steps. It is important to note that this malicious embedding neighbor detection test is based on relative errors, and thus simply tests the consistency between the estimated and measured distances between two nodes. In particular, this test is not able to evaluate the truthfulness of a node's coordinate. Indeed, if during an embedding step a node fakes its coordinate but at the same time delays the measurement probes in a way consistent with its faked position in the coordinate space (based on the knowledge of the correspondent's coordinate, as well as its own true and fake coordinates), then the resulting relative error measured between the two nodes is reduced (see figure 5.4). Indeed, if a node at coordinate X_i fakes a coordinate X'_i, then its real distance to another node (e.g. a Surveyor) at coordinate X_j is $RTT_{i,j}$ (this RTT is the time the probe actually travels between the 2 nodes). To be consistent with its faked position, the faking node should delay the measurement probe by $||X'_i - X_j|| - ||X_i - X_j||$, so that the measured RTT, $RTT'_{i,j} = RTT_{i,j} + ||X'_i - X_j|| - ||X_i - X_j||$. The relative error that should have been measured is clearly $\frac{|||X_i - X_j|| - RTT_{i,j}|}{RTT_{i,j}}$, while the relative error measured due to the faked position is $\frac{|||X'_i - X_j|| - RTT'_{i,j}|}{RTT'_{i,j}}$. Simple substitution allows the measured relative error to be rewritten as $\frac{|||X_i - X_j|| - RTT_{i,j}|}{RTT'_{i,j}}$, which is clearly lower than the relative error that would be measured if the node did not fake its coordinate (since $\mathrm{RTT}'_{i,j} > \mathrm{RTT}_{i,j}$). Therefore, a node faking a position further away from the testing node will never fail a test that it wouldn't have failed in the first place if it wasn't faking.



$$\frac{\left|\ \|X_i - X'_j\| - RTT'_{i,j}\ \right|}{RTT'_{i,j}} \le \frac{\left|\ \|X_i - X_j\| - RTT_{i,j}\ \right|}{RTT_{i,j}}$$

Figure 5.4: Fake Coordinate and Consistent Relative Error

Consequently, to verify a node's coordinate, several such tests must be performed from vantage points (Surveyors) *surrounding* the node (see figure 5.6). In this case, a node could easily fake its coordinate and consistently delay probes so that it moved away from some Surveyors without being noticed. But such fake position would necessarily also result in the node moving closer to some other Surveyors and failing the corresponding malicious embedding neighbor tests as it is next to impossible to "speed up" a distance probe protected by the simplest of mechanisms (e.g. hashing, simple encryption, random probe numbers, etc). A node must be surrounded by at least one more Surveyors than there are dimensions in the space.

If the malicious embedding neighbor test is negative at each Surveyor chosen to surround the node (i.e. the relative error observed between the Surveyor and the node is considered normal), then the coordinate claimed by the node is considered correct. Note that this test is different from a normal "triangulation" approach, where the measured distances between the node and the Surveyors would be used alongside the

Surveyors' coordinates to determine the node's own coordinate. Indeed, our test is in fact made up of multiple, independent tests on the plausibility of the observed relative errors and provides our method with an important resistance to the triangular inequality violations (TIVs) [48, 49] that can be commonly encountered in the Internet. This is because the Kalman filters underlying the tests are calibrated during normal embedding of the Surveyors, and thus in conditions where TIVs are naturally encountered, so the system noise resulting from these TIVs is therefore implicitly taken into account in the relative error tracking. We do not claim that our test is immune to the problems caused by TIVs (and these TIVs will be responsible for some of the false positives of our test), but it is nevertheless much less sensitive to them than a geometric approach like triangulation would be.

If a node cannot be surrounded by Surveyors (i.e. there is at least one axis along which the corresponding coordinate component of the node is either greater or smaller than the corresponding coordinate component of all Surveyors), then the node's coordinate cannot be verified. This is because the node could easily fake its position by moving away from the Surveyors along this axis, without being detected, by using the technique described in figure 5.4 with each Surveyor. It is therefore important to have Surveyors placed near the coordinate space boundaries, as well as inside the space (otherwise part of the nodes population may never be able to get coordinate certificates). More precisely, the convex hull[3] of all nodes (including the Surveyors) should be composed of Surveyors only. Or seen differently, any node lying outside the convex hull of the set of Surveyors can never be surrounded and thus can never be issued a coordinate certificate. For this study, and to reflect a plausible deployment scenario, some of the Surveyors were deliberately placed near the space boundaries, so that the Surveyors' convex hull encloses most, but not all, nodes, while the other Surveyors were chosen at random (see Figure 5.5).

Protocol

A node who wishes to have its coordinate certified contacts the known closest Surveyor to its claimed coordinate. If this Surveyor is not the closest, the node is redirected to the closest one known by the Surveyor. For this, as well as the selection of Surveyors

[3] The convex hull of a set of points is the smallest convex set containing all the points. Think of it as being the smallest "enclosure" that can be formed with some of the points and that has no points outside it [73].

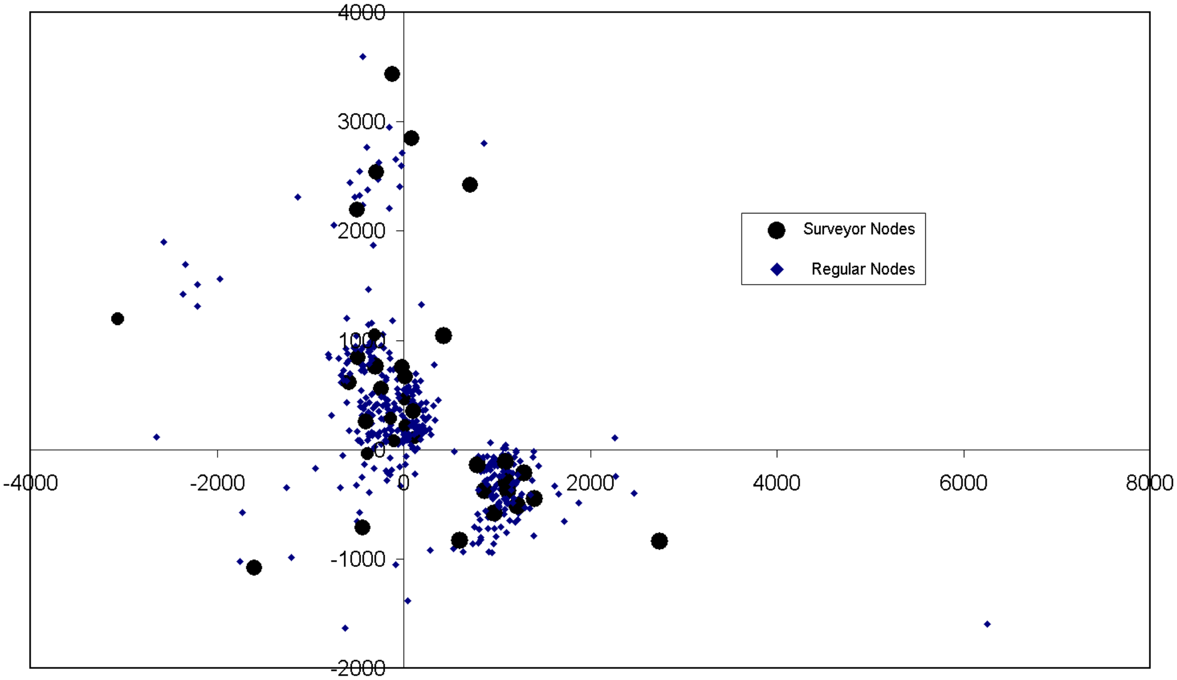


Figure 5.5: Distribution of regular and Surveyor nodes in a 2d space (distances are in ms).

surrounding the coordinate claimed by a node to happen, Surveyors exchange their coordinates using a gossiping protocol.

Based on its knowledge of the position of other Surveyors, as well as on the coordinate of the node to be certified, the certifying Surveyor selects a set of Surveyors (a.k.a surrounding Surveyors) that surround the node's claimed position (possibly including itself) and informs these of the node's claimed coordinate so to ensure they all use the same node's coordinate for their distance estimates during their malicious embedding neighbor detection test.

Recall that Surveyors compute their own coordinate by using each other exclusively as embedding neighbors. This gives the Surveyors the view of a clean system without malicious insider attacks. Therefore, if Surveyors run their own malicious embedding neighbor detection test at each embedding step, all such tests should ideally be negative as Surveyors only have trusted and honest neighbors (other Surveyors). Unfortunately, no test is perfect and some amount of the tests carried out by each Surveyor will wrongly identify the neighbor as malicious. Such occurrence is a false positive and the Surveyor will take no action about the said neighbor (or more precisely the particular embedding step with this neighbor). However, carrying out such tests at every

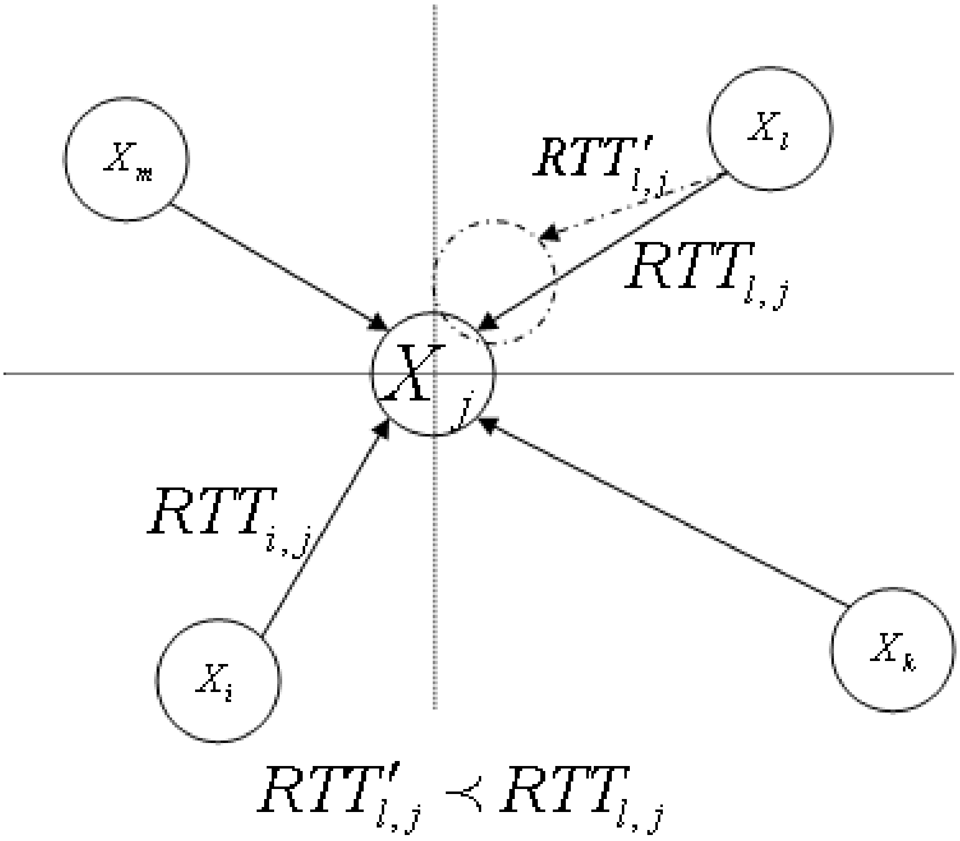


Figure 5.6: Surveyors surrounding a node for coordinates verification

embedding step provides the Surveyors with estimates of two important test statistics: the false positive test ratio (FPTR – i.e. the percentage of the tests that were positive and wrongly identified a fellow Surveyor as malicious) and the true negative test ratio (TNTR – i.e. the percentage of the tests that were negative and thus correctly identified the fellow Surveyors as honest nodes).

Let Y_i be the indicator random variable that represents the outcome of a malicious embedding neighbor detection test at the ith Surveyor (testing another Surveyor), with:

$$Y_i = \begin{cases} 0 & \text{if the neighbor identified as honest} \\ 1 & \text{if the neighbor identified as malicious} \end{cases}$$

Taking as null hypothesis H_0 that the tested node is honest (which is always the case when Surveyors are tested), the true negative test ratio (TNTR) estimate p_i at the ith Surveyor is $\mathbb{P}\text{rob}\{Y_i = 0|H_0\}$, the number of tests that were correct divided by the overall number of tests carried out. The FPTR is then obviously $1 - p_i$.

After performing the requested malicious embedding neighbor detection test on the node whose coordinate is to be certified, the surrounding Surveyors return the result of their test, along with their estimated TNTR, to the certifying Surveyor. If every test returned is negative (i.e. each surrounding Surveyor considered the node as honest), then the node's coordinate is deemed correct and verified and the certifying Surveyor proceeds to the second step of the certification described in section 5.4.2.

On the other hand, if at least one of the surrounding Surveyor returns a positive test, that is, did consider the node as potentially malicious because of too much of a deviation between the measured relative error and the expected one, the certifying Surveyor must decide whether to declare the node's coordinate as suspicious and thus refuse to issue a certificate, or whether further test should be carried out. To do so, the probability that the node, and its claimed coordinate, have been identified mistakenly as suspicious by the surrounding Surveyors is computed. This probability is simply $1 - \prod_{i \in \xi^j} p_i$, where ξ^j is the set of surrounding Surveyors chosen to verify the claimed coordinates of the node at this round of testing. If the overall probability that the node has been mistakenly classified as suspicious is greater than a given significance value γ, that is if $\prod_{1 \leq j \leq \mathcal{N}} (1 - \prod_{i \in \xi^j} p_i) > \gamma$, where $\mathcal{N}$ is the number of test rounds that have been carried out so far, then the certifying Surveyor starts another test round with a new set of surrounding Surveyors. Note that the sets of selected surrounding Surveyors at each round are not necessarily disjoint, although such property is desirable.

In this study, we used $\gamma = 1\%$ and limited $\mathcal{N}$ to 6 (i.e. a node is refused a coordinate certificate if the probability of mistaken refusal falls below 1% and/or the node fails 6 consecutive test rounds).

Evaluation

In this section, we seek to evaluate the effectiveness of our proposed coordinate verification method.

We first seek to validate the assumption that the FPTR values measured during embedding at Surveyors provide good estimates for the real FPTR values at these Surveyors. To do so, we let a vivaldi system, without cheat, converge and run on PlanetLab for over 2500 time ticks (i.e. embedding periods). the Surveyors measured their estimated FPTR by carrying out a malicious embedding neighbour detection test at every embedding step. At the end of the experiment, each Surveyor also measured its real

FPTR by running a malicious embedding neighbour detection test to every other nodes in the system. Since this system does not have any malicious node in it, a failed test is a false positive. The CDF of the differences of these 2 values at each Surveyor is shown in figure 5.7. We see that the difference between the estimated and the real FPTR are mostly within less than 1% of each other, confirming that our proposed estimation method yields reliable FPTR estimates. Even in the cases where the FPTR estimates differ more than the real value, these will only affect the coordinate verification tests in which the corresponding Surveyors take place: in these cases the coordinate verification test will be slightly more aggressive than it ought to (since the FPTR estimate is smaller than the real value), favouring security.

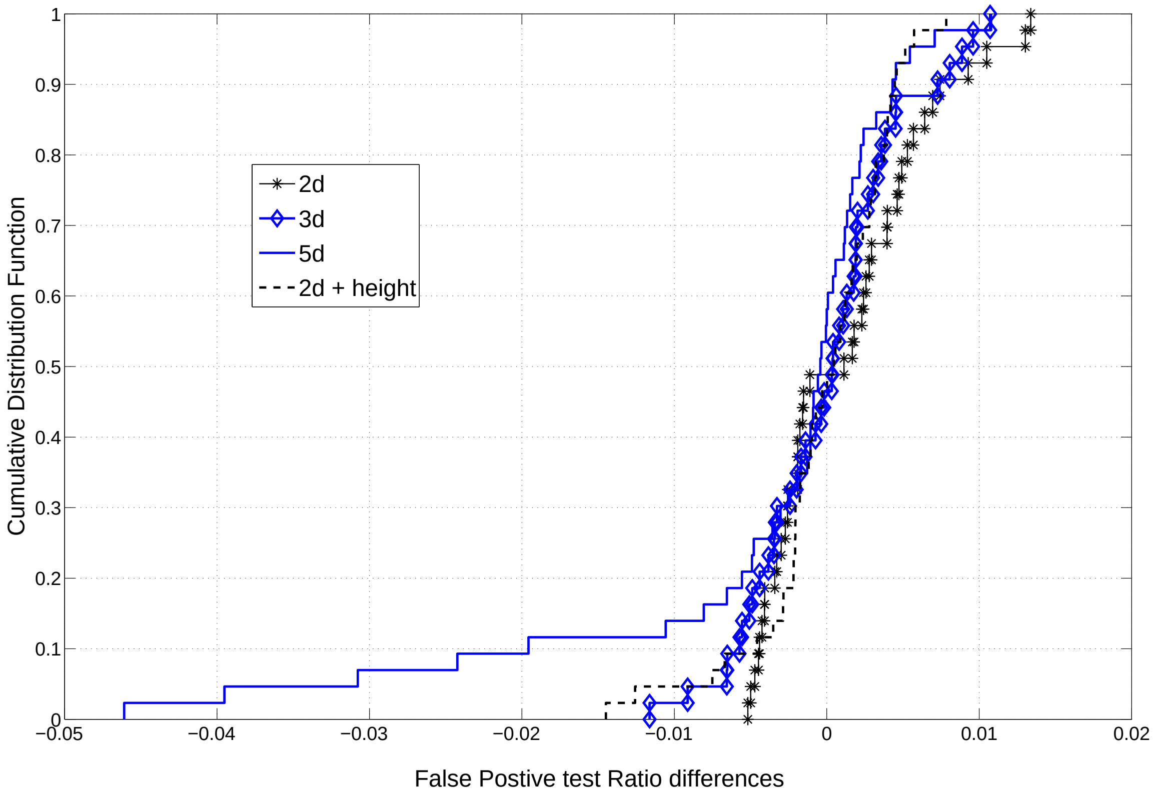
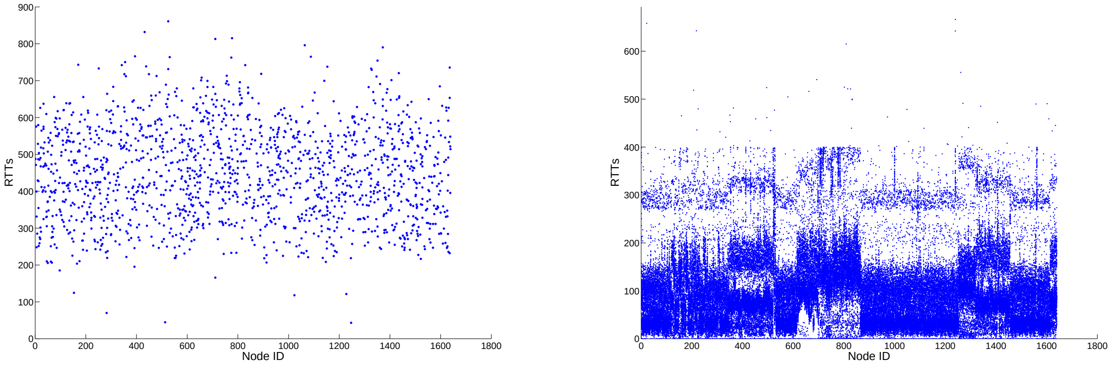


Figure 5.7: CDF of False positive probability differences.

Next, we seek to further understand the behaviour of false positive and true negative occurrences. Figure 5.8(a) is a scatter plot of the distance between a node and its Surveyor, whenever the malicious embedding neighbour detection test through a false positive at the Surveyor. This figure clearly indicates that Surveyors hardly ever experience wrong test results when testing nearby honest nodes. Complementarily, Figure 5.8(b) shows that Surveyors are much more successful at identifying nearby honest nodes correctly. These results indicate that striving to choose surrounding Surveyors as close as possible to the node whose coordinate are being verified will increase the effectiveness of the coordinate verification test (by reducing the occurrences of false positive in the multiple test, through reduction of false positive occurrences in the component tests making this multiple test up). This is therefore the strategy adopted in the rest of this chapter.



(a) Correlation between False Positives and RTTs (b) Correlation between True Negatives and RTTs

Figure 5.8: Correlation between distance and test performance.

To assess the overall effectiveness of the proposed coordinate verification method, we then make all nodes try to move away in all direction, from their position in the converged system. In this experiment, each node varies the direction of its move one degree at a time. Along each direction, it tries to move a distance drawn at random form the distribution of the innovation process computed by the Kalman filter calibrated at the nearest Surveyor, in a bid to outsmart the malicious embedding neighbour detection test. After each move, the node requests a re-certification of its coordinate. The displacement achieved by the node until the corresponding fake coordinate is declared incorrect by our verification method is then recorded. Figure 5.9 shows, the average, minimum and maximum displacement observed across all nodes, as a function of the direction of movement. Note that this figure only take into account the nodes that are eligible for certification (i.e. the nodes that lie inside the Surveyors convex hull, see section 5.4.1), meaning that 13 nodes are excluded from the experiment.

We see that the maximum displacement achieved by any node is below 10 ms,

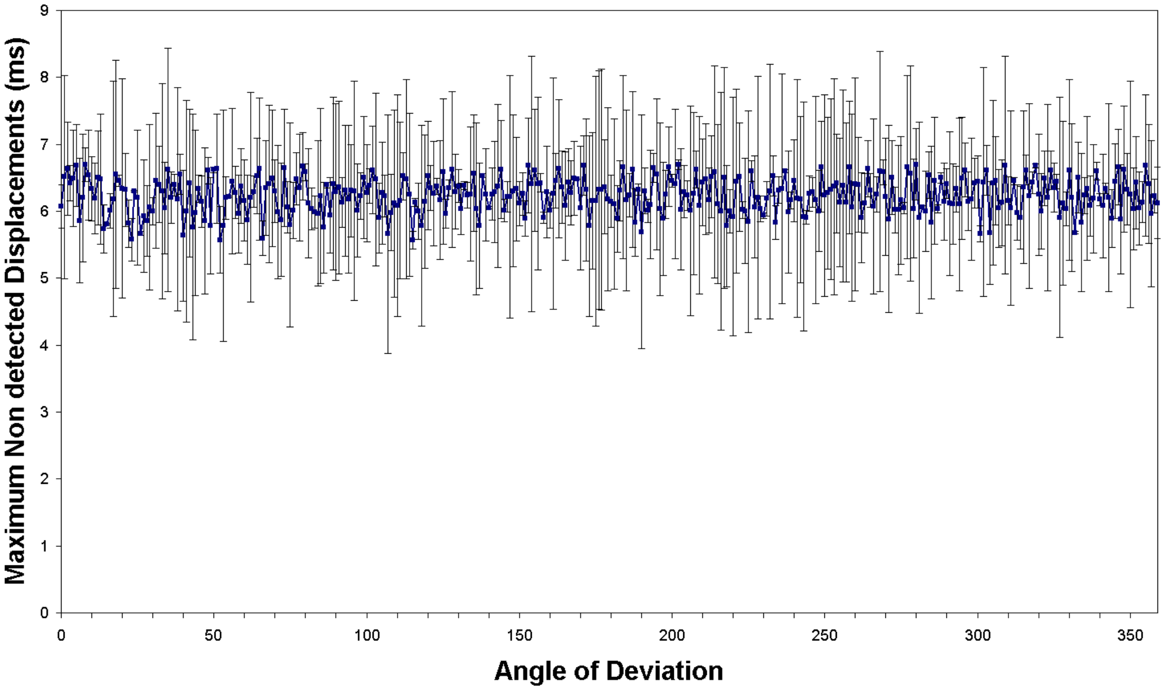


Figure 5.9: Undetected displacement as a function of direction.

while the average is around 6 ms. Such deviations are of the order of the distance prediction accuracy (they will not decrease the accuracy of the coordinate system in any significant way) and will be acceptable to many applications.

Finally, we experimented with a simple attack, carried out by a growing malicious node population that has access to the coordinates of all nodes in the system. The malicious nodes compute the centroid of the overall node population, and then try to move in a random direction away from this centroid (adding 2 seconds) in order to be isolated. Figure 5.10 shows the detection rate, that is the percentage of certificate requests for faked coordinates that was denied, as a function of the malicious population size, for various dimensions of the coordinate space. Note that although these curves show a slightly decreasing trend, a smaller percentage of an increasing number of requests for faked coordinates does mean, in most cases, an increasing number of denials. With over 95% detection rates in most cases, the coordinate verification test can be considered as highly efficient.

However, tests may achieve high detection rates by being overly aggressive towards honest nodes that do not cheat. The false positive rate, that is the percentage of certificate requests for real coordinates that were wrongly denied, is a measure of this

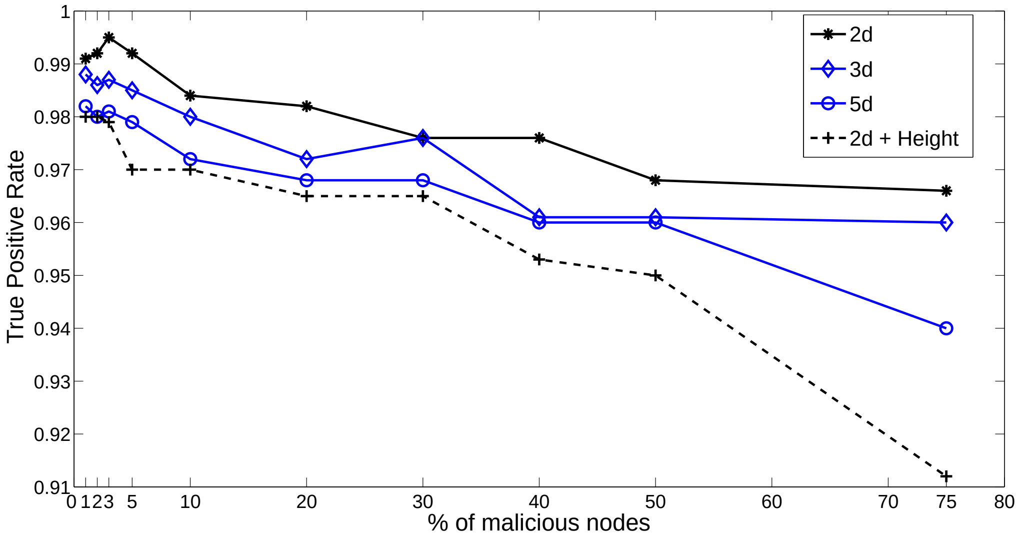


Figure 5.10: Self isolation: Detection probability.

aggressivity towards non malicious nodes and should be kept as low as possible. Figure 5.11 shows the measured false positive rate, as a function of the size of the malicious population, for various dimensionality of the coordinate space. Note that these curves depend on the performance of the test only, and not on the activities of the malicious nodes. Also, note that as the population of malicious nodes increases, the corresponding population of honest nodes decreases, so an upward trend actually corresponds to fewer wrongly denied certification requests to honest nodes. With a false positive rate lower than 6% in all cases, our test can be considered as moderately, and acceptably, aggressive.

In light of the evaluation results presented in this section, we conclude that our proposed test for coordinate verification exhibits good performance and is fit for purpose.

5.4.2 Certificate Validity Computation

After the correctness of a node's advertised coordinates has been asserted, the next step is to issue the node with a certificate proving that the enclosed coordinate has been verified. This certificate will be delivered by the certifying Surveyor (i.e. usually the Surveyor closest to the node in the coordinate space, see sections 5.3.4 and 10).

As a coordinate certificate is associated with a particular position in the coordinate space that the node occupies or has occupied, one could expect that nodes request new

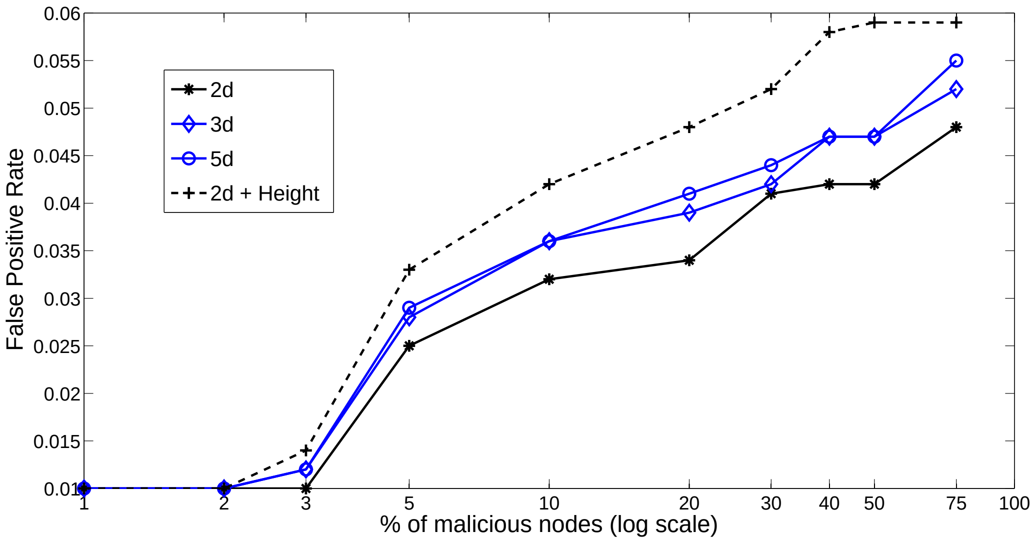


Figure 5.11: Self Isolation: False Positive Ratio.

certificate on moving to a new position. While this is certainly the behaviour anticipated from honest nodes that are interested in being able to prove their true position, malicious nodes might decide not to do so. Indeed, a malicious node is probably much more interested in faking its true position, and doing so with a certificate "proving" a position which isn't really its own, whatever this coordinate might be, is probably a bonus for such a node. While the coordinate verification protocol described in the previous section has been designed to prevent, as much as possible, malicious nodes from acquiring certificates for fake coordinates that they may choose, the problem of nodes using certificates for positions that have meanwhile changed, is still to be addressed.

The obvious way to solve this "stale" certificate problem, is to assign a reliability to the certificate. The reliability of the certificate decreases with time from its issuance time. When it crosses a certain threshold p_{th} the certificate should be invalidated and eventually reissued. There is a tradeoff between certificate precision (and therefore security) and frequency of coordinate certification that is controlled by the reliability p_{th}. Using larger value of p_{th}, leads to higher reliability for certificates but at the cost of frequently reissuing certificates that are still valid. On the other hand lower p_{th} results in lower reliability, but also reduces the load on Surveyors who would receive certificate requests less frequently. The issue here is really to find the right trade-off between scalability (by limiting the rate or amount of certificate requests) and security (by limiting the time lapse during which a malicious node may be able to use an old certificate after its coordinate has changed). We will therefore seek to exploit the results on coordinate inter-shift times presented in section 5.3.3 to compute certificate validity periods.

Note that this section assumes that all nodes in the system are “loosely” time synchronized: since coordinate inter-shift times have been shown to take values that are usually measured in minutes (see section 5.3), as long as all clocks are synchronized with an accuracy exhibiting a smaller timescale (say a few seconds), the notion of time in the system can then be considered unique. This time synchronization can easily be obtained through using NTP (at least about surveyors).

Principle

The problem of computing a validity period for coordinate certificates can be formalized through reliability theory. Let’s define the survival function of the coordinate of node i as the probability, $S^i_{T_0}(\Delta) = \mathbb{P}\text{rob}\{T^i > T^i_0 + \Delta\}$, that the next change of coordinate occurs later than Δ time units after the last coordinate change, happening at time T^i_0. The survival function is thus related to the inter-shift time cumulative distribution $F^i(\Delta) = \mathbb{P}\text{rob}\{T^i \leq T^i_0 + \Delta\}$ through $S^i(\Delta) = 1 - F^i(\Delta)$.

Recall from section 5.3.4, that the inter-shift times observed at a Surveyors are similar to those observed at nearby nodes. Hence, the inter-shift time distribution at a certifying Surveyor is the distribution used to compute the validity of the certificates it issues (since it issues certificates to the nodes that are closest to it than to any other Surveyors).

The survival function can be used to compute the validity of a certificate, that is the time remaining until the next coordinate change. The probability that the next position change occurs at or before time $\tau + \Delta$, given that the certificate is being issued at time τ is just:

$$\begin{aligned}\mathbb{P}\text{rob}\{T < \tau + \Delta | T > \tau\} &= \frac{\mathbb{P}\text{rob}\{\tau < T < \tau + \Delta\}}{\mathbb{P}\text{rob}\{T > \tau\}} \\ &= 1 - \frac{S^i(\tau + \Delta - T^i_0)}{S^i(\tau - T^i_0)}\end{aligned}$$

In fact, we use the above probability, computed at a Surveyor whose survival function and last coordinate change time are $S^i(\Delta)$ and T^i_0 respectively, to estimate the lapse of time until the next position change of the node requesting the certificate. In other words, the assumption here is that network conditions for nodes that are close to each other should change in a synchronous way. However, due to the asynchronous nature of embedding steps at different nodes, their respective coordinates will not all change at the same time, but we take as "reference" time, the moment when the Surveyor is expected to see a change in its coordinate.

In section 5.3.3, we showed that most nodes follow a lognormal or Weibull distribution. Depending on which distribution each surveyor node is observing (computing the likelihood at each embedding step), the above formula have a simple form for these two distributions.

For lognormal inter-shift distribution we will have:

$$
\begin{aligned}
S(\Delta) &= 1 - \Phi(\frac{\ln \Delta}{\sigma}) \\
\mathbb{P}\mathrm{rob}\big\{T < \tau + \Delta | T > \tau\big\} &= 1 - \frac{1 - \Phi(\frac{\ln(\tau+\Delta-T_0^i)}{\sigma})}{1 - \Phi(\frac{\ln(\tau-T_0^i)}{\sigma})}
\end{aligned}
$$

where the function $\Phi(.)$ is the complementary error function and σ is its shape parameter of the lognormal distribution. For Weibull distributions we will have:

$$
\begin{aligned}
S(\Delta) &= 1 - exp(\Delta^{\gamma}) \\
\mathbb{P}\mathrm{rob}\big\{T < \tau + \Delta | T > \tau\big\} &= 1 - \frac{1 - exp((\tau + \Delta - T_0^i)^{\gamma})}{1 - exp((\tau - T_0^i)^{\gamma})}
\end{aligned}
$$

where γ is the shape parameter of the Weibull distribution.

The validity time of a certificate, Δ, is then computed by setting $\mathbb{P}\mathrm{rob}\big\{T < \tau + \Delta | T > \tau\big\} = p_{th}$, where p_{th} is the chosen reliability of the estimate (i.e. the long-term proportions of validity periods that will expire before the corresponding coordinate change).

The certificate then consists in the verified coordinate, the timestamp of the certificate creation, the validity period Δ of the certificate, as well as the identification of the node the certificate is delivered to (i.e. IP address). The certificate is then signed by the the certifying Surveyor using appropriate credentials and encryption keys and issued to the node.

Evaluation

To evaluate the effectiveness of our certificate validity computation, we study the over-estimation resulting from each certificate: for each certificate issued, if the corresponding node moves before its current certificate expires, we record the residual time left on the certificate. This over-estimation is an important security parameter, as it is the lapse of time a malicious node could use a “stale” certificate.

Figure 5.12 shows the CDF of over-estimation times when the certificates are issued by either the closest Surveyor or a Surveyor chosen at random. We can clearly see that most certificates will not outlive their corresponding coordinates by more than 2 embedding periods in the case where they are delivered by the closest Surveyor. This is a good result, as many a time, coordinate changes in such timescale will be “localized” in space. The figure also confirms that the accuracy of the validity periods, as thus the security of the system, is improved if coordinates are certified by nearby Surveyors.

Scalability is also an important factor for any certification scheme. Although under-estimation of the validity period of certificates does not pose any security issue for the system, it does tend to increase the load on the system, and on the Surveyors in particular. Obviously, the higher the probability threshold used to compute the certificate validity time, the shorter this time will be. We therefore measure the mean validity period over all certificates for various values of the probability threshold p_{th}. Figure 5.13 shows the corresponding average certification rate, which is the inverse of the mean validity period. The average certification rate gives the average number of certification requests that will be issued per node and per time unit (here the embedding period) to the system. This number, multiplied by the number of nodes in the system, and divided by the number of Surveyors and the embedding period gives a certificate request rate per second at each Surveyor.

Figure 5.13 shows that the average certification rate increases gently as the probability threshold increases (i.e. as the computation becomes more conservative). This behaviour shows that we can afford a reliability of 95% (and therefore high security) with moderately low overhead.

5.5 Distance Estimation Performance Evaluation

In this section, we study the impact of an attack on the accuracy of distance estimations, with and without our proposed coordinate certification defense mechanism.

The attack consists in malicious nodes choosing, once the Vivaldi system has stabilized, a target they wish to get closer to (each malicious node chooses a target at random amongst the honest nodes), and moving in a straight line towards this target by steps computed from the innovation process of their Kalman filter (see chapter 4,

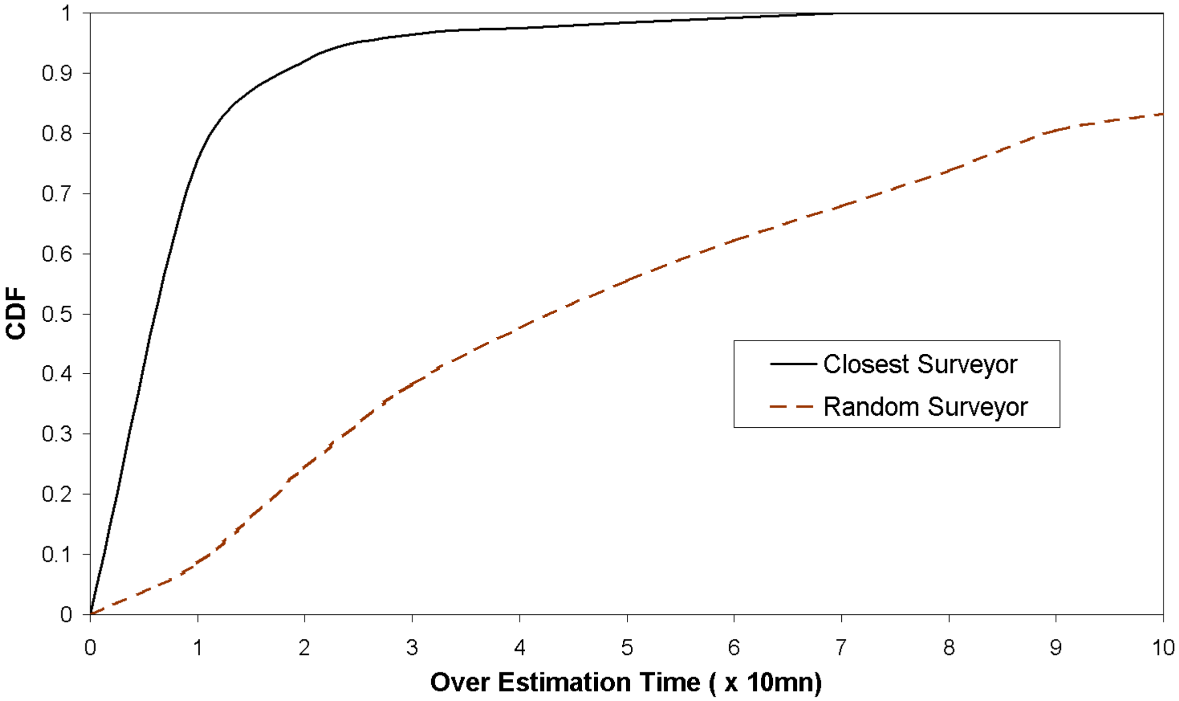


Figure 5.12: CDF of Over Estimation Times ($p_{th} = 0.95$), Embedding period = 10mn

section 4.3.1). After each displacement, the malicious node seeks to have its newly faked coordinate certified. This strategy is designed to outsmart the tests used for coordinate verification, by only taking (fake) steps that could be attributed to normal system dynamics, as well as caused by normal noise, by the detection test described in the previous chapter. We choose such subtle attack, as more obvious displacement caused by fake coordinate would be easier to detect.

Obviously, in the case where coordinate certification is employed, malicious nodes are limited to fake coordinates they can get a certificate for.

We carry out this attack on our PlanetLab experimental setup, with a varying population of malicious nodes.

To assess the impact on distance estimation, we define the following metrics:

- Relative Estimation Error $RER = \frac{|\|C_i' - C_j\| - \|C_i - C_j\||}{\|C_i - C_j\|}$, where C_i is a node's real coordinate in the system without malicious activity, and C_i' is a node's advertised coordinate (and C_i' is either faked, certified or both).
- Security Gain Ratio $SGR = meanRER_{on}/meanRER_{off}$, where $meanRER_{on}$ ($meanRER_{off}$ resp.) is the average RER measured between all pairs of node

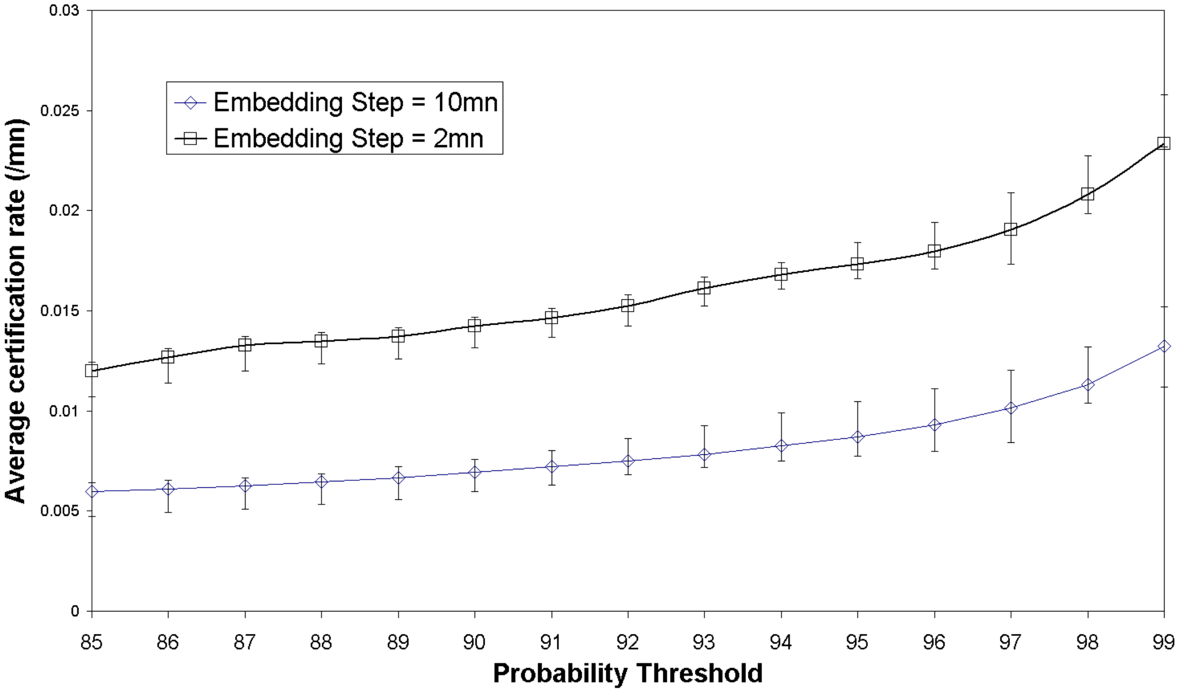


Figure 5.13: Average Certification Rate

when the security mechanism is on (off resp.).

Figure 5.14 shows the SGR observed at the end of the experiment that was allowed to run for a considerable time after convergence of the original (clean) Vivaldi system. The curve labeled “ratio1” depicts the SGR measured in the presence of malicious nodes, while the curve labeled “ratio2” depicts the SGR measured in the clean Vivaldi system without malicious activity.

From this figure, we can conclude that the accuracy of distance estimation in the system with malicious nodes is much improved when coordinate certification is in use than when it is not. This is because the coordinate verification phase of the certification filters out most of the faked displacements. We also see that the curve “ratio2” exhibits a value of 1, indicating that the presence of the certification system does not degrade the performance of the clean system without malicious nodes (in other words, the coordinate certification is very much non intrusive).

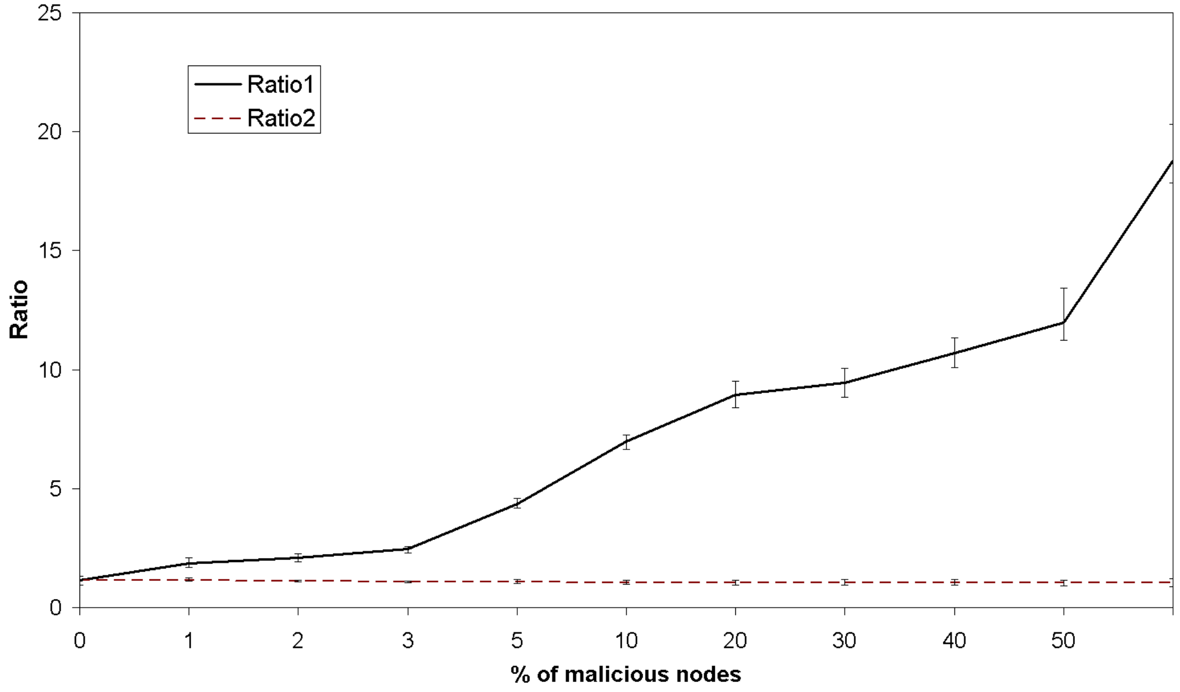


Figure 5.14: Progressive Displacement Attack: Security gain ratio varying the malicious nodes percentage in the overall population.

5.6 Conclusions

In this chapter, we have presented a coordinate certification method to protect the distance estimation phase of Internet Coordinate Systems from nodes lying about their coordinate. This work thus complements our detection protocol for the embedding phase security proposed in the previous chapter.

The proposed certification method is based on a coordinate verification method, along with a validity period estimation for the corresponding certificate. This method has been shown to be effective, enjoying good verification test performance (high true positive detection rate with low false positive rates), while achieving a very good trade-off between scalability and security. Indeed, the validity periods of certificates are rarely over-estimated, while they still do not trigger too frequent re-certifications.

Although this work focused on Vivaldi for measurements and experimentations, the method proposed for coordinate certification is independent of the embedding protocol used. This because the malicious embedding neighbour detection test that forms the basis of the coordinate verification is itself independent of the specifics of the embedding protocol, and because the validity period computation only depends on observed coordinate inter-shift times. Our proposed method would then be general enough to be applied in the context of coordinates computed by other Internet coordinate systems than Vivaldi.

6

CONCLUSIONS AND FUTURE WORK

Internet coordinate systems (e.g., [11, 12, 13, 14, 15, 16], etc.) embed latency measurements amongst samples of a node population into a geometric space and associate a network coordinate vector (or coordinate in short) in this geometric space to each node, with a view to enable accurate and cheap distance (i.e. latency) predictions amongst any pair of nodes in the population. Extensive measurements and analysis from a live, large-scale deployment have shown network coordinate systems to be fit for purpose [18], making them a valuable tool to support distributed applications, systems and overlays (e.g., [19, 10, 9]) that rely on, and benefit from, the notion of network topology-awareness. However, these coordinate systems only achieve desirable accuracy, robustness, stability and scalability properties at the expense of rather slow convergence times – in other words, a new node joining the coordinate system may not reach an accurate value for its own coordinate before several tens of seconds or even several minutes. Such convergence properties argue in favor of a deployment of Internet coordinate systems as an always-on, large-scale service.

In this thesis, we first studied in chapter 3 various types of attacks on two prominent coordinate system proposals. One of our salient findings is that larger systems are consistently more resilient than smaller ones. Given the observation in [15] and [14] that larger systems are more accurate and the well known fact that larger systems converge slower at start-up time, there seems to be a compelling case for large-scale coordinate systems to be built as a virtual infrastructure service component. The paradox is of course that always-on, large scale systems supporting many different applications will always attract more attacks than systems with a smaller reach, while the large size of the system itself would act as a particularly good terrain to create especially virulent propagation of the attack.

Our results show that there is an intrinsic trade-off to be made between accuracy and vulnerability. Indeed, we have shown that the more accurate the system for a given system size, the more susceptible it was to a same proportionate level of attack.

We have also observed that while an attack is in full swing, the performance of the coordinate systems (and of the applications it supports) can easily degrade below that of a system where coordinates are chosen randomly, whilst the aftermath of an attack could have very long lasting effects on the system due to a small number of remaining malicious nodes.

Infrastructure-based systems can also, under some well chosen attack strategies, be as vulnerable than those based on the peer-to-peer paradigm. Furthermore, the security mechanisms that have been proposed to date to defend against malicious nodes are clearly rather primitive and still in their infancy and definitely cannot defend against all types of attacks.

In the perspective of designing a detection protocol or a security mechanism, these results guide us to track the normal evolution of the coordinate-based systems, with a dimension-less metric (i.e. relative error), in order to make the technique generally applicable to different coordinate systems, and less-sensitive to different parameters we studied in chapter 3.

Therefore, in a second step, we addressed ways to secure such coordinates-based systems in a general and effective way. In chapter 4, we first show that the dynamics of a node, in a coordinate system without abnormal or malicious behavior, can be modeled by a Linear State Space model and tracked by a Kalman filter. Then we show, that the obtained model can be generalized in the sense that the parameters of a filter calibrated at a node can be used effectively to model and predict the dynamic behavior at another node, as long as the two nodes are not too far apart in the network.

In practice, we introduced in chapter 4 the concept of Surveyor nodes which, by design, are immune to embedding attacks and do observe the properties of the coordinate system in clean conditions. The Surveyors make up the basis of a “security infrastructure”. It is important to note that the deployment of Surveyors does not equate to imposing an embedding infrastructure: peer-to-peer based embedding systems, like Vivaldi, do retain their infrastructure-less embedding characteristics. Indeed, apart from a test to accept or filter out embedding steps, our method does not entail any change to the operations of the embedding protocols.

The operations of the proposed detection protocol were deliberately kept simple and tested on systems where Surveyors were chosen randomly, although their representativeness increases with closeness to their “clients”. Despite the possibly non-optimal Surveyor distribution resulting from such a choice, the results obtained show the effectiveness of our proposal in securing Internet coordinate systems. Nevertheless, given the enhanced coordinate service afforded by our detection test and simple detection protocol, one might envisage that ISPs may readily want to deploy Surveyors within their network to offer enhanced coordinate service to their customers. Such business-driven strategic deployment can only improve representativeness of Surveyors, and thus improve the security of large-scale coordinate service, with possibly a much smaller proportion of Surveyors than the upper bound reported in this thesis (as illustrated by the simple k-means deployment in chapter 4, section 4.4.3). More sophisticated Surveyor selection mechanisms than those presented in this thesis could also result in better security through better representativeness in large-scale coordinate systems.

Furthermore, each node using malicious behavior detection does so in isolation, as there is no cooperation between nodes in a bid to improve detection and identification of malicious nodes. Instead, any embedding step identified as malicious by a node is simply, and quietly, ignored and discarded locally. One of the reasons the detection protocol was designed in this way was to avoid potential denial-of-service attacks that could result from the sharing of information about malicious activity, with the view of excluding offending nodes. Indeed, such an approach could open the door to an attack that consists in trying to get honest nodes excluded from the system through the collusion of wrong malicious reports. Trust propagation could be used to mitigate or remove this threat and thus allow detection cooperation amongst nodes, which could only improve the security of the overall system and push the boundary of applicability of the detection by reducing further the impact of very large-scale attacks on the embedding system.

Considering attacks directly against the surveyor infrastructure is also one of our future directions. Indeed, there are at least two types of attacks that could be performed against the Surveyors infrastructure. The first one is a direct Denial of Service attack, where Surveyors would be flooded for instance. Since the surveyors are known, a more subtle attack, could also consist in a a large group of attackers sending enough traffic on the link between surveyors, causing queuing on those links (link clogging), to make the surveyors filters become mis-calibrated. Ultimately, a mis-calibration means a non-representativeness of the Surveyor infrastructure, and thus less accurate positioning of nodes in the coordinate-based system, or even more false positives in our detection protocol.

Even though this thesis does not address the problem of such external attacks on the infrastructure (e.g. denial of service attacks, “link clogging”, etc.), we note that solutions to such attacks have been proposed in several other works (e.g. [66, 67]). Studying the impact and ways to integrate different mechanisms against attacks that target the Surveyors themselves is then still to be considered in our detections mechanisms.

In chapter 5, we also proposed a general coordinate certification method to protect the distance estimation phase of Internet Coordinate Systems from nodes lying about their coordinate. This complements our work on the embedding phase security, in the mission of securing Internet coordinate-based positioning systems.

Our method has been shown to be effective against sophisticated attacks, and to achieve a very good trade-off between scalability and security. However, we should note that a node knows when its next embedding will occur, and thus when its coordinate is likely to change. A malicious node could exploit this knowledge to seek to obtain a certificate (for its current coordinate) just before performing this embedding, as this could leave such node in possession of a certificate for a soon to be outdated coordinate. To palliate this problem, one could envisage that Surveyors carry out “spot check” on the validity of a node’s certificate: if the certified coordinate fails a new coordinate verification test, the node is “penalized” for using an outdated certificate. Performing new tests integrating the ideas discussed above in our detection protocols, is an essential future work that is planished for the next few months.

Ultimately, we would like to end up with a practically user-friendly Secure coordinate-based service. Regardless of the accuracy of Internet coordinate systems, we believe that securing such systems is a necessary condition to their deployment. Indeed, without security of both the embedding and distance estimation phases, simple attacks have been shown to easily and seriously disrupt these systems and reduce their utility to next to nothing. We believe then that the increased robustness provided by the proposals presented in this dissertation, could act as a catalyzer to the acceptance and deployment of large-scale coordinate services in the Internet.

Appendix A

Hypothesis testing and Goodness of fit tests

In chapters 4 and 5, we used the Hypothesis testing technique for our anomalous behavior detection method. We also validate and assess our assumptions (either normality of the system error W_n in chapter 4 or the log-normal distribution of Inter-Shift distributions in chapter 5), using goodness-of-fit tests (kolmogorov-Smirnov, Lilliefors and Chisquare tests). In the following we give a description of this statistical background.

A.1 Hypothesis Testing

We consider a binary hypothesis testing problem, where there are two hypotheses, H0 (often called the null hypothesis) and H1 (also called alternative hypothesis). For each observation, the decision process must choose either hypothesis that best describes the observation.

Given two hypotheses, there are four possible outcomes when a decision is made.

The decision is called a detection when the algorithm selects H1 when H1 is in fact true. On the other hand, if the algorithm chooses H0 instead, it is called false negative. Likewise, when H0 is in fact true, picking H1 constitutes a false positive. Finally, picking H0 when H0 is in fact true is termed nominal.

The outcomes of a decision making process, and their corresponding probability notations are listed in table A.1.

Table A.1: Possible outcomes of a decision-making process

Outcome	Probability	Description
Detection	P_D	Pr[choose H1 | H1 is true]
False Positive	P_F	Pr[choose H1 |H0 is true]
False negative	$1 - P_D$	Pr[choose H0 |H1 is true]
Nominal	$1 - P_F$	Pr[choose H0 |H0 is true]

The detection probability, P_D, and the false positive probability, P_F, are typically used to specify performance conditions of any detection algorithm. In particular, for user-selected values α and β, we desire that:

$$P_F \leq \alpha \ \ and \ \ P_D \geq \beta \tag{A.1}$$

where typical values might be $\alpha = 0.01$ and $\beta = 0.99$.

According to the Neyman-Pearson paradigm [74], a decision as to whether or not to reject H_0 in favor of H_1 is made on the basis of a test statistic (typically a single number) computed on the empirical data and the fitted model. The set of values of the test statistic for which H_0 is accepted and rejected are called, respectively, the acceptance region and the rejection region. We reject the hypothesis if the test statistic

(TS) is larger than a number called the critical value (CV). Hence the rejection region is defined by $TS > CV$, and the value CV separates the rejection region from the acceptance region. One of the mistakes we can make when deciding whether or not to accept H_0 is the following: we reject H_0 when it is true. Let α denote the probability that we make this kind of mistake; α is called the significance level of the test. The rejection region (and hence critical value) is defined for a given probability α under the null hypothesis.

A.2 Goodness of fit test

A statistical test in which the validity of one hypothesis is tested without specification of an alternative hypothesis is called a goodness of fit test. The general procedure consists in defining a test statistic, which is some function of the data measuring the distance between the hypothesis and the data (in fact, the badness of fit), and then calculating the probability of obtaining data which have a still larger value of this test statistic than the value observed, assuming the hypothesis is true. This probability is called the size of the test or confidence level. Small probabilities indicate a poor fit. High probabilities (close to one) correspond to good fits.

The most common tests for goodness of fit are the chi-square test, Kolmogorov Smirnov test, Cramer-Smirnov-Von-Mises test and the Lilliefors test. In this thesis we are using different goodness-of-fit tests to validate our assumptions, namely the Kolmogorov Smirnov and the Lilliefors tests. Next we descibe these goodness of fit tests.

A.2.1 The Kolmogorov Smirnov test

For each potential variable T_i from the data set, the one-sample Kolmogorov-Smirnov test (K-S) [72]compares the proportion of values less than T_i with the expected number predicted by the distribution under test. Its function uses the maximum difference over all T_i values as its test statistic. Mathematically, this can be written as :

$$max(|F_m(T_i) - G(T_i)|)$$

where m denotes the total number of sample points, $G(x)$ the fitted cumulative distribution function, M_x the number of sample points less than x and $F_m(x) = M_x/m$.

A.2.2 The Lilliefors test

Lilliefors test is a modification of the K-S test, first presented by Lilliefors in [63]. The Lilliefors test evaluates the hypothesis that the sample has a normal distribution with unspecified mean and variance against the alternative hypothesis that the sample does not have a normal distribution. The main difference from the well-known Kolmogorov-Smirnov test (K-S test) is in the assumption about the mean and standard deviation of the normal distribution. The K-S test assumes the mean and standard deviation of the population normal distribution are known; Lilliefors test does not make this assumption. In the analysis of empirical data, more often than not the mean and variance of the population normal distribution are unknown, and must be estimated from the data. Hence Lilliefors test is generally more relevant than the K-S test.

The Lilliefors test statistic is computed from the maximum vertical offset of the empirical cdf's of (1) the sample, after conversion to Z-scores, and (2) the standard normal distribution. A sample is converted to Z-scores by subtracting the sample mean and dividing by the sample standard deviation, such that the mean of the Z-score series is 0 and the standard deviation is 1.0. Basically, considering a random sample $T_1, T_2 \ldots T_n$ of size n, which might be an observed time series. the Z-scores are computed as follows:

$$Z_i = \frac{T_i - \bar{T}}{s} \quad i = 1 \ldots n$$

where $\bar{T}$ is the sample mean, and s is the sample standard deviation.

The empirical cdf of this Z-score series is computed. Similarly, the cdf of the standard normal distribution is obtained at the same probability points. The maximum difference of the two cdf's at any point is then computed. Superimposing plots of the two cdf's immediately reveals where the cdf's differ most, and this is the point yielding the Lilliefors statistic. The Kolmogorov-Smirnov test is identical to the Lilliefors test except that no conversion to Z-scores is made in the K-S test.

A.3 conclusion

Each test statistic described above provides an indicator of the quality of the fitting of the specified distribution, and is thus called a goodness-of-fit indicator. This indicator reports a measure of the deviation of the fitted distribution from the input data: hence the smaller its value, the better the achieved fitting. To answer the question "how small a value is needed for a good fit?" one uses the critical values. If the test statistic is less than the critical value, then we accept the hypothesis. The critical value is defined for a specific significance level α. If we accept the hypothesis for a given critical value associated with α, it means that we believe there is only an α percent chance that we are making a mistake using this decision criteria; in other words, we are $1 - \alpha$ confident in our decision. The final metric we use to assess a fit is the p-value which is an indicator of the probability of a good fit; the p-value can be defined as the $Pr(TS > c)$ for some value c. If $c > CV$, then $p < \alpha$. Put simply, the p-value is the smallest value of α for which the null hypothesis will be rejected.

Appendix B

The Expectation Maximization method

The expectation-maximization (EM) algorithm is used for finding maximum likelihood estimates of a distribution's parameters, when the model depends on unobserved variables [62].

EM operates by alternatively computing an expectation of the likelihood (E step), including the unobserved variables as if they were observed, and by maximizing the expected likelihood found on the E step (M step). The parameters computed in the M step are then reused in the E step, and the process is repeated.

Let x be the set of incomplete data, consisting for example in the set of observed values and z is the missing data (we dot not observed yet for instance). Together, z and x form the complete data.

We denote the probability density function (continuous case) or probability mass function (discrete case) of the complete data with parameters given by the vector θ, as $p(x, z|\theta)$.

An EM algorithm iteratively improves an initial estimate θ_0 by constructing new estimates θ_1, θ_2, and so on. An individual re-estimation step that derives θ_{n+1} from θ_n takes the following form:

$$\theta_{n+1} = \arg\max_{\theta} Q(\theta)$$

where $Q(\theta)$ is the expected value of the log-likelihood. In other words, we do not know the complete data, so we cannot say what is the exact value of the likelihood, but given the data that we do know (the x's), we can find a posteriori estimates of the probabilities for the various values of the unknown z's. For each set of z's there is a likelihood value for θ, and we can thus calculate an expected value of the likelihood with the given values of x's (and which depends on the previously assumed value of θ because this influenced the probabilities of the z's). Q is then given by $Q(\theta) = E[\log p(x, z|\theta) \mid x]$, where it is understood that this denotes the conditional expectation of $\log p\,(\mathbf{x}, \mathbf{z}|\theta)$ being taken with the θ used in the conditional distribution of z fixed at θ_n. In other words, θ_{n+1} is the value that maximizes (M) the conditional expectation (E) of the complete data log-likelihood given the observed variables under the previous parameter value.

Appendix C

Présentation des Travaux de Thèse en Français

C.1 Introduction

Ces dernières années ont vu l'énorme croissance d'applications Internet (comme PASTRY [6], OCEANSTORE [7], ESM [8], SKYPE [9], etc.), basées et/ou bénéficiant des réseaux de recouvrement tenant compte de la topologie Internet. En particulier, la plupart des ces applications (si ce n'est toutes) et leurs réseaux de recouvrement (overlay) associés, se basent sur la notion de proximité réseau, typiquement définie en termes de délais d'aller-retour (RTT), pour l'optimisation de la sélection de voisins. Cependant, les mesures de proximité, peuvent s'avérer extrêmmement coûteuses en termes de consommation en bande passante. En effet, l'existence simultannée de plusieurs réseaux de couvertures, peut entraîner un surcoûts de communications élevé, dû aux mesures de proximité individuelles menées par chaque noeud du réseau de couverture. De plus, traquer la proximité au sein d'un groupe dynamique, nécessite une fréquence de mesure très grande. Cela induit encore plus de surcoûts de mesures.

Afin de pallier ce problème, les systèmes de positionnement Internet, comme [11, 12, 13, 14, 15, 16], ont été introduits. Dans ces systèmes, l'idée principale, est que si chaque nœud peut être associé à une coordonnée virtuelle dans un espace approprié, la distance entre les nœuds est trivialement calculée sans pour autant avoir recours aux mesures directes. En d'autres termes, ces systèmes plongent les mesures des temps de latences (délais) entre une population de nœuds dans un espace géométrique et associent un vecteur de coordonnées (ou coordonnées) dans cet espace à chaque nœud, dans le but de permettre des prédictions de distances précises et peu onéreuses parmi n'importe quelle paire de nœud dans le réseau. L'avantage premier de ces systèmes, est que si les distances réseaux (dans le sens de délais ou de temps de latence) sont plongées dans un espace de coordonnées, où une position raisonnablement précise pour chaque nœud est établie, le surcoût de mesures produit par le positionnement, est ainsi amorti sur plusieurs prédictions de distances. Ceci réduit énormément le coût en terme de mesures de distances du système entier.

Des mesures et analyses approfondies de déploiements réels de ces systèmes ont montré qu'ils atteingnaient pleinement leurs objectifs [18], faisant d'eux un outil précieux pour supporter les applications distribuées et réseaux de recouvrement (tels que [19, 10, 9]), qui bénéficient de la notion de proximité réseau. En effet, les systèmes de positionnement à base de coordonnées atteignent plusieurs propriétés souhaitables, telles que la précision, la robustesse, la stabilité, ainsi que le passage à l'échelle et les surcoûts de communications peu élevés. Cependant, il est important de souligner que ces propriétés sont souvent réalisées aux dépends de temps de convergence assez longs. Dans de tels systèmes, les nouveaux nœuds, n'atteignent une bonne estimation de leurs coordonées, qu'après un laps de temps à l'échelle de dizaines de seconde, voire de minutes. Ceci est beaucoup trop lent, comparé aux temps de convergence réalisés avec des mesures directes entre les nœuds, et peut même être inacceptable pour certaines applications, qui ont pour but de rapidement identifier les “meilleurs nœuds”. Nous soutenons donc, dans cette thèse, l'idée que les systèmes de positionnement Internet, en particulier ceux basés sur les coordonnées, sont une proposition attractive s'ils sont déployés comme un service : chaque nœud fait tourner un système de coordonées au démarrage de son système d'exploitation. Cela permet ainsi au nœud de fournir des estimations de coordonnées précises à la demande des applications et réseaux de couvertures. Un système de coordonnées est alors perçu comme une “infrastructure virtuelle” qui supporte une large variété d'applications et réseaux de couverture. Cela soutient l'idée d'un déploiement des systèmes de coordonnées en tant que service à grande échelle.

C.1.1 Motivations, Problématiques et Contributions

Du fait de l'utilisation de plus en plus répandue des réseaux Pair à Pair et de recouvrement, se basant sur les systèmes à base de coordonnées, on peut aisément imaginer la propagation de vers et autres programmes malicieux dont le but est d'attaquer ces systèmes. En particulier, il est interéssant de noter qu'un système fournissant un service à large échelle, serait aussi une principale cible pour les pirates informatique, puisque sa perturbation mènerait au disfonctionnement ou à l'effondrement de plusieurs applications.

Le fait que les systèmes de coordonnées actuels supposent que les nœuds évoluant dans le système coopèrent entièrement et honnêtement les uns avec les autres, et que cela peut aussi les rendre vulnérables aux attaques, a été l'une de nos motivations premières pour aborder le problème de sécurité dans ce type de systèmes. En particulier, les attaques internes lancées par des nœuds (potentiellement en collusion) infiltrant le système, pouvent s'avérer très dangereuses.

Notre première contribution a été donc d'étudier à quel point ce danger est réel pour les systèmes de coordonnées. Nous avons ainsi identifié trois types d'attaques potentielles contre les systèmes de positionnement. Spécifiquement, nous avons étudié comment ces attaques mènent à des imprécisions dans les prédictions de distances. Nous avons montré qu'il est assez facile de réaliser une attaque de déni de service (DoS), en fournissant des coordonnées biaisées ou en retardant les mesures lancées par les nœuds voisins. Nous avons aussi analysé des moyens simples permettant à des nœuds malicieux de prendre le contrôle du système, puisqu'ils sont capables d'imposer à d'autres nœuds honnêtes, des positions dans le réseau, sans pour autant être détectés. Enfin, nous avons étudié la manière dont une conspiration d'attaquants est réalisée, et combien elle affecte ce genre de systèmes. L' efficacité de ces attaques contre leurs systèmes cibles, a été démontrée à travers des simulations approfondies de deux systèmes représentatifs : Vivaldi et NPS.

Un de nos principaux résultats a été que les systèmes plus grands sont plus robustes que ceux impliquant un petit nombre de nœuds. L' observation dans [15] et [14], qui stipule que les systèmes plus larges sont plus précis et le fait que ces mêmes systèmes à grande population convergent moins rapidement au démarrage, consolide donc encore plus le fait que ces systèmes soit construits comme une infrastructure virtuelle offrant un service de large échelle. Le paradoxe est bien évidemment qu'un service étendu, supportant plusieurs applications, attire toujours plus d'attaquants. Nos résultats ont aussi montré qu'il y a un compromis, intrinsèque au système, entre précision et vulnérabilité. En effet, nous avons montré que plus le système est précis, plus il est vulnérable pour la même proportion d'attaquants dans le système.

Nous avons aussi montré que les systèmes de coordonnées à infrastructures fixes, sont, sous des stratégies d'attaques bien choisies, aussi vulnérables que les systèmes basés sur le paradigme pair-à-pair. Nous démontrons finalement, que les mécanismes de sécurité proposés jusque là pour spécifiquement défendre certains systèmes contre les nœuds malicieux, sont encore primitifs, et ne peuvent absolument pas protéger le dit système contre tous les types d'attaques.

Ainsi, guidés par notre compréhension des mécanismes d'attaques et de leurs conséquences sur les systèmes des coordonnées, nous nous sommes, par la suite, posés la question : Comment sécuriser le processus de positionnement dans les systèmes à base de coordonnées?

Notre contribution majeure a été donc tout naturellement de proposer un protocole de détection et de sécurisation des systèmes de positionnement à base de coordonnées. En constatant qu'en l'absence de nœuds malicieux, les coordonnées d'un nœud dépendent de la combinaison des conditions réseau et de la spécificité du processus de calcul des coordonnées lui-même (exemple : quel type de protocole est utilisé, quel est le nombre de dimensions de l'espace géométrique choisi, etc), nous avons introduit le concept de nœuds experts (Surveyors). Les nœuds experts forment un groupe de nœuds de confiance (honnêtes), dispersés dans le réseau, qui se positionnent en utilisant exlusivement d'autres nœuds experts. En même temps, ces nœuds aident les autres nœuds 'normaux' pour leur positionnement. Cette stratégie permet ainsi aux nœuds experts d'apprendre l'évolution naturelle de l'espace de coordonnées, en observant l'évolution de leurs propres coordonnées, en l'absence d'activité malicieuse. L'idée est que les nœuds experts partagent une représentation du comportement normal dans le système avec les autres nœuds afin de leur permettre de détecter et de filtrer les comportement anormaux.

Nous affirmons et vérifions que, en absence d'activité malicieuse, les coordonées d'un nœud peuvent être vu comme un processus stochastique à dépendances linéaires, dont l'évolution pourrait être traquée par un filtre de Kalman [4]. Chaque nœud expert calcule et calibre les paramètres d'un modèle linéaire, charactérisant l'évolution des ajustements de ses coordonnées, et partage les paramètres de ce modèle avec d'autres nœuds. Ceux-ci utilisent donc ces paramètres, pour faire tourner leurs propres filtres de Kalman. La sortie des filtres, en l'occurence le processus d'innovation, est utilisée pour comparer les ajustements de coordonnées tels qu'observés avec ceux prédits par le filtre de Kalman. Ainsi on considère comme suspect, les étapes d'ajustements de coordonnées, où la différence serait trop élevée. Une combinaison de simulations et expérimentations PlanetLab ont été utilisées pour démontrer la validité, la généralité et l'efficacité de l'approche proposée pour chacun des systèmes Vivaldi et NPS.

Nous nous sommes par la suite intéressés au problème de sécurité de la phase d'estimation des distances dans de tels systèmes. En particulier, le problème de la validité des coordonnées Internet telles qu' annoncées par les nœuds d'un système de coordonnées durant la phase d'estimation des distances. En effet, certains nœuds peuvent délibérément mentir quant à la valeur exacte de leurs coordonnées afin de lancer diverses attaques contre les applications et les réseaux de couverture.

La méthode proposée se divise en deux étapes : 1)établir l'exactitude des coordonnées annoncées en utilisant l'infrastructure des nœuds experts et la méthode de détection des nœuds malicieux, et 2) délivrer un certificat à validité limitée pour chaque coordonnée vérifiée. Les périodes de validité sont calculées à partir d'une analyse des temps d'inter-changement observés par les nœuds experts. En faisant cela, chaque nœud expert, peut estimer le temps jusqu'au prochain changement de coordonnées, et ainsi, peut limiter le temps de validité du certificat qu'il délivrerait aux nœuds normaux. Notre méthode est illustrée en utilisant une trace recueillie à partir d'un système Vivaldi déployé sur PlanetLab, où les distributions de temps d'inter-changements suivent des distributions longue traine (distribution lognormale dans la plupart des cas, et distribution Weilbull sinon).

C.1.2 L'organisation de la thèse

Nous commençons cette dissertation en faisant, dans le Chapitre 2, un tour d'horizon des systèmes de positionnement de l'Internet, existant dans la littérature, ainsi que de leurs exploitations potentielles dans diverses applications. Nous décrirons notamment notre proposition de réseau multipoint se basant sur un processus de localisation, afin de permettre le passage à l'èchelle [35, 34]. Dans ce chapitre, nous nous concentrons néanmoins sur la description des différents systèmes à base de coordonnées existants, ainsi que sur leurs mécanismes de sécurité.

Dans une deuxième étape, nous identifions dans le Chapitre 3 diverses attaques contre les systèmes de positionnenement à base de coordonnées, et montrons l'impact que peuvent avoir de telles attaques sur les performances de ces systèmes [1, 2].

Nous avons par la suite proposé une méthode générale pour la détection des comportements malicieux au sein des systèmes de positionnement durant la phase de calcul des cordonnées.

Nous montrons en premier lieu, dans le Chapitre 4, que la dynamique d'un nœud, dans un système de coordonnées sain, exempt de comportements anormaux ou malhonnêtes, peut être modélisée par un modèle d'états linéaires, et traquée par un filtre de Kalman. De plus, les paramètres d'un filtre calibré au niveau d'un nœud donné, peuvent être utilisé pour modéliser et prédire le comportement dynamique d'un autre nœud, tant que ces deux nœuds sont proches l'un de l'autre dans le réseau.

Cela a entraîné la proposition d'une infrastructure de nœuds, appelés "nœuds experts" : des nœuds de confiance, se positionnant dans l'espace des coordonnées, en utilisant exclusivement d'autres noeuds experts. Ils sont ainsi immunisés contre des comportements malicieux dans le système. Pendant le calcul de leurs propres coordonnées, les autres noeuds peuvent ainsi utiliser les paramètres du filtre d'un nœud expert proche, comme étant une représentation d'un comportement normal, non assujetti à un comportement malicieux, pour détecter et filtrer toute activité malicieuse ou anormale. Une combinaison de simulations et d'expérimentations PlanetLab a été utilisée pour démontrer la validité, la généralité, et l'efficacité de l'approche proposée pour chacun des deux systèmes Vivaldi et NPS [3].

Dans le chapitre 5, nous sécurisons la phase d'estimation des distances dans les systèmes de coordonnées. La méthode proposée se divise en deux étapes : 1)établir l'exactitude des coordonnées annoncées en utilisant l'infrastructure des nœuds experts et la méthode de détection des nœuds malicieux, et 2) délivrer un certificat à validité limitée pour chaque coordonnée vérifiée. Les périodes de validité sont calculées à partir d'une analyse des temps d'inter-changement observés par les nœuds experts. En faisant cela, chaque nœud expert, peut estimer le temps jusqu'au prochain changement de coordonnées, et ainsi, peut limiter le temps de validité du certificat qu'il délivrerait aux nœuds normaux. Notre méthode est illustrée par une trace recueillie à partir d'un système Vivaldi déployé sur PlanetLab, où les distributions de temps d'inter-changements suivent des distributions longue traine (distribution log-normale dans la plupart des cas, et distribution Weilbull sinon). Nous montrons l'efficacité de notre méthode en mesurant l'impact de plusieurs attaques sur les estimations de distance, expérimentées sur PlanetLab.

C.2 Chapitre 2: Une vue d'ensemble des systèmes de positionnement Internet

Dans ce chapitre, nous présentons une vue d'ensemble des différentes propositions dans le domaine des systèmes de positionnement Internet. Nous commençons par décrire quelques travaux destinés à fournir une estimation de la localisation et de la proximité dans le réseau. Ces systèmes ne reposent pas cependant sur les coordonnées virtuelles. Nous les appelerons "Services d'estimation de proximité par mesures directes". Nous nous concentrons par la suite sur les systèmes à base de coordonnées, que nous classifions en deux classes principales: Les systèmes basés sur les balises (landmarks), et les systèmes distribués.

En particulier, nous étudierons les systèmes munis de mécanismes de sécurité déstinés à filtrer les nœuds malicieux. Nous montrerons cependant dans la suite de cette dissertation, que ces mécanismes sont encore primitifs, et ne peuvent en aucun cas défendre le système contre tous les types d'attaques.

C.2.1 Services d'estimation de proximité par mesures directes

Plusieurs approches dans la littérature fournissent des estimations de distances ou de proximité réseau en utilisant des mesures directes entre les paires de nœuds. Dans cette section, nous présentons quelques uns de ces systèmes les plus connus. Nous nous intéressons en premier lieu aux approches de géolocalisation, qui s'inscrivent dans une tentative de fournir la localisation géographique des nœuds du réseau, plutôt que leurs positions Internet. Nous décrivons en outre l'approche "Constraint-Based Geolocation" et le système IP2Geo. Puis, nous nous intéressons à l'approche de classement de mesures vers les balises ("Binning"), et à une des applications utilisant cette approche (CAN). Nous présentons brièvement comment le réseau de couverture CAN est construit en utilisant le schéma Binning. Enfin, nous exposons les approches à base de "traceroute" et l'approche Meridian, et nous nous attardons sur une des applications d'estimation de la localisation que nous avons proposé, en l'occurence LCC [34, 35].

C.2.2 Systèmes de coordonnées à base de Balises Fixes

Ces systèmes incorporent une composante centrale (un ensemble d'entités balises), dont les nœuds se servent pour calculer leurs coordonnées en fonction de mesures vers ces entités balises. Le concept de positionnement par rapport à des entités balises a été introduit dans GNP (Global Network Positioning). Dans ce genre de systèmes, les coordonnées des balises sont en premier lieu calculées en minimisant l'erreur entre les distances mesurées et les distances estimées entre les nœuds balises. De la même manière, un nœud ordinaire dérive ses coordonnées en minimisant l'erreur entre les distances mesurées et les distances estimées vers les nœuds balises. Plusieurs approches ont par la suite repris le concept des balises, et ont pour vocation de mieux passer à l'échelle (Exemple : Lighthouse et l'approche de balises virtuelles). Nous nous intéressons plus particulièrement au système NPS.

le système NPS

Ce système [14] étend le système GNP à un système de coordonnées hiérarchique, où les nœuds peuvent servir comme balises (appelés points référence) pour d'autres nœuds. Le but principal est de pallier les problèmes de failles des balises centralisées et fixes, notamment lorsque ces entités deviennent des goulots d'étranglement dans le réseau. La différence majeure par rapport à GNP est que n'importe quel nœud qui s'est bien positionné dans le réseau peut être choisi comme un point référence par un serveur pour d'autres nœuds. Cependant, pour assurer une cohérence dans le système, NPS impose un positionnement hiérarchique entre les nœuds. Étant donné un ensemble de nœuds, NPS les partitionne dans différents niveaux. Un ensemble de 20 balises fixes sont placées dans le niveau 0, la couche supérieure du système, et qui définit la base de l'espace géométrique choisi. Chaque nœud dans un niveau L_i, choisit au hasard quelques nœuds dans le niveau L_{i-1} comme ses points référence. L'erreur relative de la prédiction de distance entre une paire de nœuds est définie comme :

$$erreur\ relative = \frac{|reelle - virtuelle|}{\min(reelle, virtuelle)}$$

Nous nous intéressons plus particulièrement au système de sécurité inclus dans NPS, qui vise à atténuer les attaques de nœuds malicieux. En effet, les nœuds peuvent mentir sur leurs positions et influencer les mesures effectuées. L'idée principale est d'éliminier le point référence s'il mène à une erreur relative trop élevée par rapport aux autres points référence. Dans cette thèse, nous considèrerons NPS, comme le système représentatif des systèmes de coordonnées à base de balises.

C.2.3 Les Systèmes de coordonnées décentralisés

Cette classe de systèmes étend le concept de positionnement par coordonnées, en généralisant le rôle des balises à tous les nœuds du système, ou en éliminant l'infrastructure des balises. Ces systèmes peuvent être alors perçus comme des systèmes de positionnement pair-à-pair. Nous introduisons en premier lieu le système PIC (Practical Internet Coordinates) où tous les nœuds du système peuvent jouer le rôle de balises pour les autres. Le système de sécurité proposé par PIC, se base sur l'inégalité triangulaire afin de détecter les nœuds malicieux, qui d'après PIC, sont plus suscpetibles de violer cette inégalitée. Dans [13], les auteurs montrent que ce test de sécurité peut pallier les attaques d'intensité allant jusquá 20% de nœuds malicieux dans le système. Cependant, [49] et [48] ont montré que les RTTs dans le réseau violent régulièrement et couramment les inégalités triangulaires. Un système de sécurité basé sur le fait que l'inégalité triangulaire est persistante, pourrait mener à une dégradation des performances du système de coordonnées, notamment, lors de l'inexistence de nœuds malicieux.

Nous nous focalisons dans ce chapitre sur le système Vivaldi, et nous le considérerons dans cette thèse, représentatif des Systèmes de coordonnées décentralisés.

Vivaldi

Vivaldi [15] est un système de coordonnées complètement décentralisé. Il est basé sur une simulation de ressorts, où la position du nœud correspond à celle de l'extrémité d'un ressort qui minimiserait l'énergie potentielle des ressorts, et donc minimiserait l'erreur de positionnement. Un nœud se joignant au système, calcule ses coordonnées en collectant des informations de positions et de mesures de délai à partir de quelques autres nœuds. Spécifiquement, Vivaldi place un ressort entre chaque paire de nœuds (i, j) dans le système, avec une longueur "de repos" correspondant à la mesure RTT entre ces deux nœuds. La longueur réelle du ressort est considérée comme étant l'estimation de distance entre les deux positions des nœuds. L'énergie potentielle d'un tel ressort est proportionnelle au carrée de déplacement par rapport à sa longueur "de repos". La somme de ces erreurs à travers tous les ressorts est la fonction d'erreur que Vivaldi tente de minimiser. Une procédure identique tourne sur tous les nœuds Vivaldi. Celle-là est basée sur des échantillons récoltés par les nœuds qui fournissent les informations pour leur positionnement. Un échantillon, utilisé par un nœud i est ainsi constitué de la mesure vers un nœud j, de la coordonnée du même nœud j, ainsi que de l'erreur locale reportée par j. L'algorithme traîte les nœuds à haute erreur, en assignant des poids à chaque échantillon collecté. L'erreur relative de cet échantillon, e_s, est alors calculée comme suit :

$$e_s = | \parallel x_j - x_i \parallel - \mathrm{RTT}_{mesure} \mid / \mathrm{RTT}_{mesure}$$

Le nœud calcule alors le poids de cet échantillon comme étant $w = e_i/(e_i + e_j)$, où e_i est l'erreur actuelle (locale) du nœud i. Ce poids est utilisé en fait pour calculer un déplacement adaptatif, δ définissant la fraction de mouvement que le nœud est autorisé à faire en direction du nœud auquel il se mesure : $\delta = C_c \times w$, où C_c est une constante < 1. Le nœud met alors à jour sa coordonnée locale comme suit :

$$x_i = x_i + \delta \cdot (\mathrm{RTT}_{mesure} - \parallel x_i - x_j \parallel) \cdot u(x_i - x_j)$$

où $u(x_i - x_j)$ est un vecteur unitaire donnant la direction de déplacement de i. Enfin, le nœud met à jour son erreur locale comme étant $e_i = e_s \times w + e_i \times (1 - w)$.

C.2.4 Discussion

Qu'ils soient basés sur des balises fixes, ou qu'ils soient décentralisés, la plupart des systèmes de positionnement actuels réalisent de bonne performances en termes de précision de positionnement, de stabilité et de passage à l'échelle. Cependant, il faut aussi noter que cela est permis aux dépends de temps de convergence longs, allant de quelques secondes à plusieurs minutes [18]. Cela est très loin des performances en termes de rapidité de traitement des systèmes de mesures directes entre des paires de nœuds, et ceci est de surcroît non accpetable pour les applications tenant compte de la topologie et qui visent à déterminer les “meilleurs nœuds”, le plus rapidement possible.

Les systèmes de positionnement à base de coordonnées sont une proposition attractive s'ils sont déployés comme un service : chaque nœud pourra alors faire tourner un système de coordonées au démarrage du système d'exploitation. Cela pourra ainsi permettre au nœud de fournir des estimations de distance à la demande, aux applications et réseaux de recouvrement. Un système de coordonnées est ainsi vu comme une “infrastructure virtuelle” qui supporte une large variété d'applications et réseaux de couverture.

Cela dit, un système fournissant un service à large échelle, serait aussi une principale cible pour les pirates informatique, puisque sa perturbation pourrait mener au disfonctionnement ou l'effondrement de plusieurs applications à la fois. Le fait que les systèmes de coordonnées actuels supposent que les nœuds évoluant dans le système coopèrent entièrement et honnêtement les uns avec les autres, et que cela peut aussi les rendre vulnérables aux attaques, a été l'une de nos motivations premières pour addresser le problème de sécurité dans ce type de systèmes. En particulier, les attaques internes lancées par des nœuds (potentiellement en collusion) infiltrant le système, pourrait s'avérer très dangereuses. De ce fait, sécuriser la base de la prédiction de distance pour plusieurs applications serait plus critique, que de détailler les artefacts d'une quelconque sécurité d'une application particulière. Dans le chapitre suivant, nous commencerons ainsi par montrer qu'il est assez simple de s'attaquer au service de coordonnées, et nous détaillerons les principaux moyens pour y arriver, ainsi que leurs impacts sur les systèmes à base de coordonnées.

C.3 Chapitre 3: Les Attaques contre les systèmes de coordonnées et leurs impacts

Ce chapitre a pour principal but d'identifier les différentes attaques contre les systèmes de coordonnées et de montrer leur efficacité à travers leur expérimentation sur deux systèmes représentatifs : Vivaldi et NPS. Notre étude basée sur des simulations montre ainsi qu'il est assez simple de déstabiliser ces systèmes et cela même si quelques mécanismes de sécurité existent. En particulier, nous quantifions les effets de stratégies d'attaques qui visent à (i) introduire du désordre dans le système, (ii) tromper des nœuds honnêtes afin de leur imposer des coordonnées loin de leur positions correctes et (iii) isoler certains nœuds cibles à travers des collusions de nœuds malicieux.

C.3.1 Menaces et classification des attaques

Nous considérons des nœuds malicieux ayant accès aux mêmes données que les utilisateurs légitimes. Cela veut dire que tous les participants ne sont pas forcément des entités de confiance, ou alors que certains nœuds malicieux ont pu contourner un quelconque mécanisme d'authentification. Les nœuds malicieux sont capables d'envoyer des informations erronées quand les autres nœuds mesurent leurs distances vers eux, ou alors d'envoyer des informations manipulées en recevant des requêtes de positionnement. Ils peuvent aussi affecter certaines métriques observées par leur cible. Les classes principales d'attaques sur les systèmes de coordonnées sont :

1. Le Désordre: le but principale d'une telle attaque est de créer le chaos sous une forme d'une attaque de déni de service. Cela entraîne des erreurs élevées dans le positionnement des nœuds, ou une non-convergence de l'algorithme de calcul des coordonnées. L'attaque consiste simplement à maximiser l'erreur relative des autres nœuds dans le système, passivement en choisissant de ne pas coopérer ou en falsifiant les coordonnées, ou alors activement en retardant les mesures effectuées par les nœuds honnêtes.

2. L' Isolation: où les nœuds cibles seraient isolés dans l'espace de coordonnées. Cette attaque cible un nœud particulier, dans le but de le convaincre qu'il est positionné dans une zone isolée du réseau. Le but ultime est, par exemple, d'obliger ce nœud cible, à se connecter à un nœud complice, étant le plus proche dans la zone isolée, pour pouvoir par la suite lancer des attaques d'analyse de traffic ou d'homme du milieu (Man in the Middle). Une manière d'aboutir à la réalisation d'une telle attaque, est de retarder les mesures envoyées par la victime, et de falsifier ses propres coordonnées, de manière à ce que la victime calcule des coordonnées plus élevées que la réalité, et s'éloigne ainsi de sa position (et par la même des autres nœuds).

3. La Répulsion: où un nœud malicieux essait de convaincre ses victimes qu'il est positionné loin d'eux afin de réduire son attractivité, et ainsi, par exemple, se soulager de la charge de retransmission de paquets, en ne coopérant pas dans les processus normaux de l'application. Une manière de réussir une telle attaque, est de faire en sorte de montrer que ses conditions (en termes de performances ou de position) sont pires qu'elles ne le sont en réalité. Cela est accompli par le biais de retardements de mesures, et/ou en manipulant les coordonnées transmis aux autres nœuds.

4. Le Contrôle du système: cete attaque est possible dans le cas où des nœuds "normaux" peuvent être considérés comme des balises, i.e. la plupart des systèmes existants exceptés les systèmes centralisés. Dans les systèmes hiérarchiques, comme NPS, les nœuds essayent de monter dans la hiérarchie dans le but de tromper ou d'influencer le maximum de nœuds honnêtes.

Évaluation des performances

Nous avons utilisé l'erreur relative (comme définie pour chaque algorithme) comme indicateur principal de performance. Comme nous nous concentrons surtout sur l'impact des nœuds malicieux sur tout le système, nous avons aussi introduit le ratio d'erreur relative comme étant l'erreur relative mesurée en présence de nœuds malicieux normalisée à la performance du système sans tricheurs. Ceci est utilisé comme le meilleur des scénarios. Une valeur au dessus de 1 indique manifestement une dégradation dans la précision du système.

Nos simulations ont montré que les systèmes de coordonnées, dćentralisées ou à base de balises, peuvent être déstabilisés, bien que nous avons aussi constaté que les systèmes plus larges sont plus résistants aux attaques. Nos résultats ont aussi montré qu'il y a un compromis intrinsèque au système entre précision et vulnérabilité. En effet, nous avons montré, en variant les dimensions utilisées par les systèmes, que plus le système est précis, plus il est vulnérable pour la même proportion d'attaquants dans le système.

Nous avons aussi pu observer que, quand les attaques tournent à plein régime, les performances des systèmes de coordonnées (ainsi que les applications qui les supportent), peuvent facilement se dégrader au delà de celle d'un système qui utiliseraient des coordonnées aléatoires. Enfin, nous avons montré que les conséquences des attaques durent très longtemps à cause des résidus d'erreurs se propageant et infectant tous les nœuds dans le système.

C.4 Chapitre 4: Sécurisation de la phase de calcul de coordonnées

Dans ce chapitre, nous nous fixons comme objectif de proposer une méthode générale et efficace pour sécuriser les systèmes de coordonnées lors de leur phase de calcul des coordonnées.

Nous montrons ainsi en premier lieu, que la dynamique d'un nœud, dans un système de coordonnées exempt de comportement malicieux ou anormal, peut être modélisée par un modèle linéaire et pistée par un filtre de Kalman.

C.4.1 Modélisation du processus de calcul des coordonnées

Le but des systèmes de coordonnées, sans tenir compte de la manière de calcul, ni de l'aspect dimensions de l'espace géométrique utilisé, est d'attribuer des coordonnées à un nœud dans le système de manière, à ce qu'à tout moment, on puisse assimiler la distance géométrique entre deux coordonnées à la distance réseau réelle, mesurée en RTT entre deux nœuds. La mesure RTT en revanche, manifestement, dépend de l'état du réseau (exemple : de la charge du trafic, de l'état des files d'attentes dans les routeurs, etc.), mais aussi de l'état des systèmes d'exploitation des nœuds eux-même générant du bruit dans les mesures. Ainsi la valeur exacte de RTT varie continuellement.

Cela se répercute sur les calculs de coordonnées effectuées par les systèmes de positionnement, en dehors du fait qu'ils soient décentralisés ou basés sur des balises fixes. À chaque étape de calcul des coordonnées, la précision de ce calcul est déterminée en observant la déviation entre le RTT mesuré vers un nœud correspondant et celui estimé dans le système de coordonnée. Plus précisemment, cette précision est définie comme étant l'*erreur relative mesurée* $D_n = \frac{|\|X_i^n - X_j^n\| - RTT_{ij}^n|}{RTT_{ij}^n}$. Le but de n'importe quel système de coordonnées est de minimiser un indicateur de coût (l'erreur moyenne quadratique) qui capture l'erreur relative mesurée.

Les erreurs relatives mesurées sont sujettes à des fluctuations des RTTs pour les raisons mentionnées ci-dessus, en particulier, les congestions transitoires dans le réseau et les problèmes d'ordonnancement dans les systèmes d'exploitation. Afin d'isoler l'impact de ces fluctuations de RTT sur la détection d'anomalie, nous introduisons Δ_n, l'*erreur relative nominale*, que notre nœud pourrait obtenir si à l'étape n de ses calculs de coordonnées les RTTs ne fluctuent pas. Une anomalie devient alors une simple observation de larges déviations de D_n, l'erreur relative mesurée, de sa valeur nominale Δ_n.

Parce que plusieurs sources contribuent à la déviation de D_n de sa valeur nominale (erreur des mesures RTT, fluctuations des RTT, erreurs dans les coordonnées des nœuds), il est raisonnable de supposer que les deux valeurs sont fonctions l'une de l'autre comme suit :

$$D_n = \Delta_n + U_n \tag{C.1}$$

où U_n est une variable gaussienne à moyenne 0 et à variance ν_U.

Nous nous focalisons maintenant sur la dynamique du système dans son régime nominal lorsque les RTTs ne fluctuent pas. Nous définissons ainsi l'*erreur du système* W_n qui représente l'impact des autres nœuds dans le système sur le positionnement du nœud considéré à l'étape n. Puisque W_n résulte de plusieurs sources contributrices, il est aussi raisonnable de supposer que c'est un processus gaussien (avec une moyenne $\bar{w}$ et une variance v_W).

Comme première approximation, le processus Δ_n peut ainsi être modélisé comme un modèle auto régressif d'ordre 1 :

$$\Delta_{n+1} = \beta\Delta_n + W_n. \tag{C.2}$$

où β est un facteur constant strictement inférieur à 1, afin que l'erreur relative puisse converger vers un régime stationnaire indépendemment des conditions initiales.

Les équations C.2 and C.1 définissent alors un modèle linéaire d'évolution de l'erreur relative d'un nœud. Notre objectif est d'obtenir des prédictions de cette erreur relative à partir de ce modèle. De par ses propriétés linéaires, le filtre de Kalman est utilisé dans ce cas pour pister l'évolution de l'erreur relative nominale et pour aussi obtenir une prédiction de cette erreur relative $\hat{\Delta}_{n|n-1}$.

En utilisant la méthode EM ("Expectation Maximization method"), nous calibrons le filtre de Kalman afin de déterminer pour chaque nœud dans un système normal, les valeurs des paramètres $\theta = (\beta, v_W, v_U, w_0, p_0)$ utilisés par le filtre.

Cela nous a mené à la proposition d'une infrastructure de noeuds experts : ceux-ci sont des noeuds de confiance, se positionnant dans l'espace des coordonnées, en utilisant exclusivement d'autres nœuds experts. Ils sont ainsi immunisés contre n'importe quel comportement malicieux dans le système. Pendant le calcul de leurs propres coordonnées, les autres noeuds peuvent ainsi utiliser les paramètres du filtre calibré d'un noeud expert non assujetti à un comportement malicieux, pour détecter et filtrer toute activité malicieuse ou anormale.

C.4.2 Validation du modèle

Une combinaison de simulations et d'expérimentations PlanetLab a été utilisée pour démontrer la validité du modèle proposé pour chacun des deux systèmes Vivaldi et NPS. Nous avons ainsi pu montrer en premier lieu la validité de nos hypothèses, en particulier la supposition que le bruit du système W_n est un processus gaussien. Ensuite, nous avons montré que les erreurs relatives telles que prédites par le filtre et celles réellement calculées par les nœuds dans le système sont très semblables. Lorsque le modèle a convergé (i.e. la méthode EM a convergé et les variations pour tous les paramètres varient d'une valeur inférieure à 0.02), nous relançons le système de coordonnées, et nous observons les erreurs de prédiction comme étant la valeur absolue entre l'erreur prédite par le filtre de Kalman au niveau d'un nœud et l'erreur relative mesurée réelle. Nous observons que les erreurs de prédiction sont la plupart du temps inférieures à 0.02.

Nous avons par la suite vérifié qu'un ensemble de nœuds de confiance déployés au hasard dans le réseau peut être représentatif de l'ensemble des nœuds dans le système. Une des premières questions que nous nous sommes posées était donc de savoir combien de nœuds de confiance (experts) fallait-il déployer pour reprśenter l'évolution des erreurs des autres nœuds. Nous avons ainsi pu montrer, qu'en effet une proportion de moins de 8% de nœuds experts aléatoirement déployés dans le système de coordonnées est suffisante pour pouvoir représenter le comportement du système.

Ayant montré que la population de nœuds experts peut représenter le comportement normal de tout le système, la question suivante que nous nous sommes posée était de savoir à quel point le comportement du système pisté par un filtre de Kalman calibré au niveau d'un nœud expert peut représenter le comportement d'un nœud normal. Nous avons observé dans nos simulations et expérimentations PlanetLab que, même si chaque nœud normal trouve au moins un nœud expert dont le filtre de Kalman mène à de très faibles erreurs de prédiction, tous les nœuds experts ne permettent pas en revanche une bonne représentativité pour un nœud normal donné. Le choix des nœuds experts est ainsi primordial pour gagner en représentativité de comportement normal, en performance de prédiction et ainsi en détection de comportements malicieux. Nos expérimentations ont alors montré qu'il existe une forte coorrélation entre les erreurs de prédiction et les RTTs entre le nœud normal et le nœud expert.

En conclusion, durant leur calcul de coordonnées, les nœuds normaux obtiennent les paramètres des filtres des nœuds experts pour pouvoir prédire correctement leurs erreurs relatives, à condition que ces nœuds experts soient positionnés proches d'eux.

C.4.3 Détection des comportements malicieux

Si les nœuds experts sont capables de calculer leurs coordonnées dans un système exempt d'une quelconque activité malicieuse, ils peuvent alors être considérés comme des entités de confiance. Puisque ces nœuds interagissent seulement entre eux mêmes, ils sont donc immunisés contre tout comportement malicieux ou anormal, et observent donc le comportement du système dans des conditions normales. Ainsi, l'idée principale de notre détection est d'utiliser le modèle de comportement normal obtenu par les nœuds auprès des nœuds experts les plus proches pour pouvoir détecter des anomalies dans le système, et ce à chaque étape de calcul des coordonnées.

C.4.4 Évaluation des performances

Nous avons évalué en premier lieu les performances de notre protocole de détection, et nous avons prouvé que les taux de faux positifs de nos tests étaient très bas, alors que les taux de vrais positifs étaient constamment élevés et ce indépendemment des dimensions de l'espace géométrique utilisé. Nous avons par ailleurs, pu démontrer à travers nos simulations et expérimentations PlanetLab, que Vivaldi et NPS se trouvaient muni d'une protection pouvant leur permettre de fonctionner normalement et ce même lorsque un peu plus de 30% de la population était maligne.

C.5 Chapitre 5: Sécurisation de la phase d'estimation des distances

Les systèmes de coordonnées sont principalement déstinés à évaluer les distances entre les nœuds, en se basant simplement sur les coordonnées échangées par les nœuds dans le système. Quelque soit le mécanisme d'échange de ces coordonnées (échanges directs, répertoire WEB, etc.), chaque nœud devrait d'une manière ou d'une autre reporter une valeur de ses coordonnées jusque là calculées. Cela crée une opportunité à des nœuds malicieux pour essayer de tricher quant à leurs coordonnées transmises pour aboutir à des fins malicieuses et à des attaques sur les applications utilisant le système de coordonnées (Déni de service, Isolation, etc.). Ce chapitre traite donc de la véracité des coordonnées transmises par les nœuds, et de la garantie de leur authenticité.

Nous proposons ainsi d'exploiter l'infrastructure de nœuds experts et du protocole de détections de triches proposés dans le chapitre précédent. En particulier, chaque nœud expert mesure sa distance vers un nœud afin de vérifier l'exactitude de ses coordonnées annoncées (en utilisant le test de détection des anomalies en erreurs relatives). Si tous les nœuds experts s'accordent à dire que les coordonnées du nœuds sont valides, on lui accorde alors un certificat, à temps de validité limitée, incluant les coordonnées certifiées.

Un temps de validité pour chaque certificat est nécessaire parce que, le changement des conditions réseau fait en sorte que les coordonnées varient dans le temps. Au changement de ses coordonnées, un nœud honnête cesserait d'utiliser son certificat actuel, et demanderait un certificat pour ses nouvelles coordonnées. En revanche, un nœud malicieux pourrait continuer à utiliser le certificat relatif à ses anciennes coordonnées. Ainsi, un compromis est nécessaire entre passage à l'échelle et validité des certificats. Une de nos contributions dans ce chapitre a été donc d'étudier les temps d'inter-changement (i.e. le temps qui sépare les changements de coordonnées d'un nœud) dans le système Vivaldi déployé sur PlanetLab. Nous avons pu observer que ces temps d'inter-changements suivaient une distribution 'lognormale' pour la plupart des cas, et une distribution Weibull dans certains cas particuliers (à noter que ces deux distributions sont des distributions longue traine).

À travers des expérimentations PlanetLab, nous avons pu montré que cette méthode est très efficace, avec un taux de détection élevé (vrai positifs) et un taux de faux positif très bas. Cette méthode permet aussi un compromis entre passage à l'échelle et sécurité. En effet, les temps de validité des certificats sont très rarement sur estimés, alors qu'ils ne déclenchent pas souvent des re-certifications.

C.6 Chapitre 6: Conclusions et Travaux Futurs

Dans cette dissertation, nous avons pu montrer dans le chapitre 3 que divers types d'attaques pouvaient être lancées contre les systèmes de coordonnées.

Nous avons alors exploré dans une seconde étape, un moyen général et efficace pour sécuriser de tels systèmes. Dans le chapitre 4, nous avons montré que la dynamique d'un nœud, dans un système de coordonné sans activité malicieuse, peut être modélisée par un modèle d'états linéaires, et traquée par un filtre de Kalman. De plus, les paramètres d'un filtre calibré au niveau d'un nœud donné, peuvent être utilisés pour modéliser et prédire le comportement dynamique d'un autre nœud, tant que ces deux nœuds sont proches l'un de l'autre dans le réseau.

Les prédictions effectuées par le filtre au niveau de chaque nœud lui permettent de comparer les erreurs relatives prédites et celles mesurées. Une déviation trop grande entre ces deux valeurs permet au nœud de ne pas considérer la mesure qu'il vient d'effectuer, le munissant ainsi d'un moyen de défense lors du calcul de ses coordonnées. Cependant, même cette sécurité reste insuffisante, lorsqu'on sait que ces systèmes sont surtout utilisés par les applications pour l’estimation des distances entre nœuds.

Dans le chapitre 5, nous avons alors abordé le problème de l’authenticité des coordonnées transmises (ou annoncées) par les nœuds. Nous avons étudié les temps d’inter-changements des coordonnées, et pu observé qu’ils suivaient souvent une distribution lognormale (dans de rares cas une distribution Weibull), et qu’un temps de validité de ces coordonnées pouvait être déterminé par un nœud expert. Celui-ci vérifie l’exactitude de la coordonnée annoncée par le nœud normal et lui délivre un certificat à temps de validité égale à la validité estimée de sa coordonnée actuelle.

Les résultats obtenus dans cette thèse, notamment pour le protocole de détection des nœuds malicieux lors du calcul des coordonnées, se sont basés sur un déploiement aléatoire des nœuds experts. Bien que la représentativité de ces derniers augmente avec un déploiement plus stratégique, ces résultats ont montré l’efficacité de notre protocole de sécurisation des systèmes à base de coordonnées. On pourrait envisager d’autre stratégies de déploiment plus optimal des nœuds experts afin d’aboutir à une meilleure représentativité à large échelle.

Considérer des attaques directes sur l’infrastructure des nœuds experts est aussi une de nos directions furtures. En effet, au moins deux attaques peuvent être envisagées contre les nœuds experts. La première serait tout simplement un déni de service par innondation des des nœuds experts. La seconde serait de s’attaquer uniquement aux liens entre certains nœuds experts, rendant ainsi les filtres au niveau de ces nœuds mal-calibrés. Une calibration biaisée est équivalente à une non représentativité des nœuds experts, résultant en positionnement incorrect ou même en un taux élevé de faux positifs dans notre protocole de détection.

Étudier l’impact de telles attaques et intégrer des méchanismes de défense contre des attaquants ciblant les nœuds experts, est aussi un de nos travaux futurs.

Nous avons aussi montré que notre méthode pour la validation des coordonnées annoncées est efficace contre des attaques sophistiquées. Cependant, il est à noter qu’un nœud peut connaître à quel moment son prochain calcul de coordonnées va avoir lieu, et ainsi quand est ce que ses coordonnées vont probablement changer. Un nœud malicieux peut chercher à exploiter cette connaissance pour obtenir un certificat (pour ses coordonnées actuelles) juste avant l’instant de son nouveau positionnement, et cela lui permettrait d’avoir un certificat valide pour des coordonnées anciennes. Afin de pallier ce problème, on peut envisager que les nœuds experts effectuent des vérifications périodiques aléatoires des coordonnées certifiées. Si ces derniers échouent le test de vérification, le nœud concerné pourrait être pénalisé, pour avoir utilisé un certificat périmé. Nous planifions de lancer de nouvelles expérimentations intégrant ces mécanismes pour évaluer de nouveau l'efficacité de notre méthode de certification.

Nous nous fixons comme objectif final de fournir un service sécurisé pratique de coordonnées. En dehors de la précision des systèmes de coordonnées, nous pensons que sécuriser de tels systèmes, est une condition nécessaire à leur déploiement. En effet, sans la sécurité de chacune des phases de calcul des coordonnées et d'estimation des distances, nous avons pu montrer que des attaques simples pouvaient réduire ces systèmes à une utilité futile. Nous croyons alors que la résistance aux attaques, que nous proposons dans cette thèse, pourrait agir comme un catlyseur au déploiment massif des services de coordonnées à large échelle.

Bibliography

[1] M. A. Kaafar, L. Mathy, T. Turletti and W. Dabbous, *Real attacks on virtual networks: Vivaldi out of tune*, In Proceedings of the SIGCOMM workshop on Large Scale Attack Defense LSAD 2006,p129-146, PISA, September 2006. 1, 34, 62, 139

[2] M. A. Kaafar, L. Mathy, T. Turletti, and W. Dabbous, *Virtual Networks under Attack: Disrupting Internet Coordinate Systems*, In Proceedings of CoNext 2006, Lisboa, December, 2006. 1, 62, 85, 86, 91, 139

[3] M. A. Kaafar, L. Mathy, C. Barakat, K. Salamatian, T. Turletti and W. Dabbous, *Securing Internet Coordinate Embedding Systems*, In proceedings of ACM SIGCOMM, Kyoto, Japan, August 2007. 1, 140

[4] R.E. Kalman, and R.S. Bucy, *New Results in Linear Filtering and Prediction Theory*, In Transactions of the ASME - Journal of Basic Engineering Vol. 83: pp. 95-107, 1961. 1, 5, 63, 138

[5] M. A. Kaafar, L. Mathy, C. Barakat, K. Salamatian, T. Turletti and W. Dabbous, *Certifying Internet Coordinates*, Under Submission, 2007. 2

[6] A. Rowstron and P. Druschel, *Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems*, In Proceedings of IFIP/ACM International Conference on Distributed Systems Platforms, Heidelberg, Germany, November, 2001. 2, 9, 135

[7] J. Kubiatowicz et al., *OceanStore: An Architecture for Global-Scale Persistent Storage*, In Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000), Cambridge, November 2000. 2, 9, 135

[8] Y. h. Chu, S. G. Rao and H. Zhang, *A case for end system multicast*, In Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS), Santa Clara, June 2000. 2, 9, 135

[9] `www.skype.com` 2, 3, 123, 135, 136

[10] Azureus BitTorrent Client. `http://azureus.sourceforce.net` 2, 3, 28, 123, 136

[11] T. E. Ng, and H. Zhang, *Predicting internet network distance with coordinates-based approaches*, In Proceedings of the IEEE INFOCOM, New York, June 2002. 2, 18, 19, 95, 123, 135

[12] M. Pias, J. Crowcroft, S. Wilbur, S. Bhatti, and T. Harris, *Lighthouses for Scalable Distributed Location*, In Proceedings of International Workshop on Peer-to-Peer Systems (IPTPS), Berkeley, February 2003. 2, 20, 123, 135

[13] M. Costa, M. Castro, A. Rowstron, and P. Key, *Practical Internet coordinates for distance estimation*, In Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS), Tokyo, March 2004. 2, 24, 25, 123, 135, 143

[14] T. E. Ng and H. Zhang, *A Network Positioning System for the Internet*, In Proceedings of the USENIX annual technical conference, Boston, June 2004. 2, 4, 7, 21, 23, 34, 58, 63, 72, 73, 123, 135, 137, 142

[15] F. Dabek, R. Cox, F. Kaashoek and R. Morris, *Vivaldi: A decentralized network coordinate system*, In Proceedings of the ACM SIGCOMM, Portland, Oregon, August 2004. 2, 4, 7, 26, 27, 34, 58, 63, 72, 73, 123, 135, 137, 143

[16] Y. Shavitt and T. Tankel, *Big-bang simulation for embedding network distances in euclidean space*, In Proceedings of the IEEE INFOCOM, San Francisco, April 2003. 2, 27, 123, 135

[17] R. Zhang, Y. C. Hu, X. Lin, and S. Fahmy,*A Hierarchical Approach to Internet Distance Prediction*, In Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS), Lisboa, July 2006. 2

[18] J. Ledlie, P. Gardner, and M. Seltzer, *Network Coordinates in the Wild*, In Proceedings of NSDI, Cambridge, April 2007. Available as Harvard University Computer

Science Technical Report TR-20-06, October 2006. 3, 28, 29, 62, 65, 100, 123, 136, 144

[19] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, *Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications*, In Proceedings of SIGCOMM, San Diego, CA, August 2001. 3, 9, 123, 136

[20] J. Liebeherr, M. Nahas, and W. Si, *Application-layer multicast with delaunay triangulations*, Tech. Rep., University of Virginia, November 2001. 9

[21] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, *A scalable content-addressable network*, In Proceedings of ACM SIGCOMM, San Diego, CA, August 2001. 9, 12

[22] A distributed Peer-to-Peer data-sharing system. The gnutella protocol specification v0.4, Available from `http://www9.limewire.com/developer/gnutellaprotocol0.4.pdf` 9

[23] BitTorrent Specification wiki. `http://wiki.theory.org/BitTorrentSpecification` 9, 28

[24] B. Cohen,*Incentives Build Robustness in BitTorrent*, In Proceedings of the Workshop on Economics of P2P Systems (p2pecon), Berkeley, June 2003. 9

[25] A. Rowstron, A. M. Kermarrec, M. Castro, and P. Druschel, *SCRIBE: The design of a large-scale event notification infrastructure*, In Proceedinsg of Networked Group Communication, NGC 2001, London, UK, November 2001. 9

[26] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida, *Constraint-Based Geolocation of Internet Hosts*, In Proceedings of ACM Internet Measurement Conference, Sicily, Italy, October 2004. 10

[27] P. Enge and P. Misra, *Special issue on global positioning system*, In Proceedings of the IEEE, 87(1):3-15, January, 1999. 11

[28] V. Padmanabhan and L. Subramanian, *An Investigation of Geographic Mapping Techniques for Internet Hosts*, In Proceedings of ACM SIGCOMM, UC San Diego, August 2001. 11

[29] P. Francis, S. Jamin, V. Paxson, L. Zhang, D. F. Gryniewicz, and Y. Jin, *An architecture for a global Internet host distance estimation service,* In Proceedings of IEEE INFOCOM, New York, NY, March 1999. 11

[30] S. Ratnasamy, M. Handly, R. Karp and S. Shenker, *Topologically-Aware Overlay Construction and Server Selection*, In Proceedings of IEEE Infocom, New York, June 2002. 11, 12

[31] S. Hotz, *Routing Information Organization to Support Scalable Routing with Heterogeneous Path Requirements*, PhD thesis, Dept. of Computer Science, University of Southern California, 1994. 13

[32] N. Spring, R. Mahajan and D. Wetherall, *Measuring ISP Topologies with Rocketfuel*, In Proceedings of ACM SIGCOMM, Pittsburgh, August 2002. 14

[33] B. Wong, A. Slivkins and E. G. Sirer, *A Lightweight Approach to Network Positioning without Virtual Coordinates*, In Proceedings of ACM SIGCOMM, Philadelphia, August 2005. 14

[34] M. A. Kaafar, T. Turletti, and W.Dabbous, *A Locating-First Approach for Scalable Overlay Multicast*, In Proceedings of IEEE IWQoS, New Haven,CT, USA, June 2006. 14, 139, 141

[35] M. A. Kaafar, T. Turletti, and W. Dabbous, *A Scalable Overlay Scheme for Application-layer Multicast*, In IEEE INFOCOM student Workshop, Barcelona, April 2006. 14, 139, 141

[36] D. A. Helder and S. Jamin, *End-host multicast communication using switch-trees protocols*, In Proceedings of GP2PC, Berlin, May 2002.

[37] Z. Li and P. Mohapatra, *Hostcast: A new overlay multicast protocol*, In Proceedings of IEEE ICC, Anchorage (Alaska), June 2003.

[38] S. W. Tan, A. G. Waters, and J. Crawford, *Meshtree: A Delay optimised Overlay Multicast Tree Building Protocol*. Techical Report 5-05, Univ. of Kent, April 2005.

[39] J Stribling,*PlanetLab All Pairs Pings*, data available from `http://www.pdos.lcs.mit.edu/~strib/pl_app/` 16

[40] S. Rewaskar and J. Kaur, *Testing the Scalability of Overlay Routing Infrastructures*, In Proceedings of of the Passive Active Measurement (PAM) Workshop, Sophia Antipolis, France, April 2004. 2, 16

[41] J. A. Nelder and R. Mead, *A simplex method for function minimization* Computer Journal, pages 308-313, 1965. 19

[42] L. Tang and M. Crovella, *Virtual landmarks for the Internet*, In Proceedings of Internet Measurement Conference, Miami Beach, FL, October 2003. 20

[43] J. Bourgain,*On Lipschitz embedding of finite metric spaces in Hilbert space*, Israel J. Math. 52, pages 46-52, 1985. 20

[44] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press, 1988. 21

[45] G. Strang,*Linear Algebra and Its Application*, Harcourt, Inc., 1988. 21

[46] H. Lim, J. Hou, and C. H. Choi, *Constructing internet coordinate system based on delay measurement*, In Proceedings of Internet Measurement Conference (IMC), Miami, FL, October 2003. 21

[47] Y. Shavitt and T. Tankel, *The curvature of the Internet and its usage for overlay construction and distance estimation*, In Proceedinsg of the IEEE INFOCOM, Hong Kong, March 2004. 27, 28

[48] H. Zheng, E. K. Lua, M. Pias, and T. Griffin, *Internet Routing Policies and Round-Trip Times*, In Proceedings of the Passive Active Measurement (PAM), Boston, March 2005. 25, 27, 94, 107, 143

[49] E. K. Lua, T. griffin, M. Pias, H. Zheng, and J. Crowcroft, *On the accuracy of Embeddings for Internet Coordinate Systems*, In Proceedings of Internet Measurement Conference (IMC), Berkeley, October 2005. 25, 27, 94, 107, 143

[50] J. Ledlie, P. Pietzuch, and M. Seltzer, *Stable and Accurate Network Coordinate*, In Proceedings of ICDCS, Lisbon, Portugal, July 2006. 28

[51] L. Peterson, T. Anderson, D. Culler, and T. Roscoe, *A Blueprint for Introducing Disruptive Change in the Internet*, In Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets), October 2002. Also available in ACM SIGCOMM Computer Communication Review, vol. 33, no. 1, January 2003, pages 59-64. 28

[52] C. de Launois, S. Uhlig, and O. Bonaventure,*A Stable and Distributed Network Coordinate System*, Technical report, Universite Catholique de Louvain, December 2004. 28

[53] Y. Bartal, N. Linial, M. Mendel, and A. Naor,*On metric ramsey-type phenomena*, In Proceedings of the Annual ACM Symposium on Theory of Computing (STOC), San Diego, CA, June 2003. 29

[54] J. Shneidman, P. Pietzuch, M. Welsh, M. Seltzer, and M. Roussopoulos,*A Cost-Space Approach to Distributed Query Optimization in Stream Based Overlays*, In Proceedings of NetDB'05, Tokyo, Japan, April 2005. 29

[55] S. J. Lee, P. Sharma, S. Banerjee, S. Basu, and F. Rodrigo, *Measuring bandwidth between planetlab nodes*, In proceedinsg of Passive Active Measurement Conference (PAM), Boston, MA, March 2005. 29

[56] D. Oppenheimer, D. A. Patterson, and A. Vahdat, *A Case for Informed Service Placement on PlanetLab*, Technical Report 04-025, PlanetLab, 2004. 29

[57] K. P. Gummadi, S. Saroiu, and S. D. Gribble, *King: Estimating Latency between Arbitrary Internet End Hosts*, In Proceedings of SIGCOMM Internet Mesasurement Workshop (IMW), Pittsburgh November 2002. 34, 73

[58] A simulator for peer-to-peer protocols. `http://www.pdos.lcs.mit.edu/p2psim/index.html` 34

[59] Y. Zhang, N. Duffield, V. Paxson, and S. Shenker, *On the Constancy of Internet Path Properties*, In Proceedings of ACM SIGCOMM Internet Measurement Workshop, San Francisco, November 2001. 64, 101

[60] Z. Ghahramani, G. Hinton, *Parameter Estimation for Linear Dynamical Systems*, University of Toronto, Technical Report CRG-TR-96-2. 67, 70

[61] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*, Prentice Hall, Ch. 7, 1993. 67

[62] A. Dempster, N. Laird, and D. Rubin, *Maximum likelihood from incomplete data via the EM algorithm*, Journal of the Royal Statistical Society, Series B, 39(1):1-38, 1977. 69, 133

[63] H. Lilliefors, *On the Kolmogorov-Smirnov test for normality with mean and variance unknown*, Journal of the American Statistical Association, Vol. 62. pp. 399-402, June, 1967. 73, 130

[64] A. Soule, K. Salamatian, and N. Taft, *Combining Filtering and Statistical Methods for Anomaly Detection*, In Proceedings of Internet Measurement Conference (IMC), Berkeley, October, 2005. 83

[65] J. Bilmes, *A gentle tutorial on the EM algorithm including gaussian mixtures and baum-welch*, Technical Report TR-97-021, International Computer Science Institute, Berkeley, CA, 1997. 71

[66] A. Keromytis, V. Misra and D. Rubenstein, *SOS: Secure Overlay Services*, In Proceedings of ACM SIGCOMM, Pittsburgh, PA, August 2002.

[67] J. Jung, B. Krishnamurthy, and M. Rabinovich, *Flash Crowds and Denial of Service Attacks: Characterization and Implications for CDNs and Web Sites*, In Proceedings of the Eleventh International World Wide Web Conference, Honolulu, Hawaii, May 2002. 126 126

[68] L. Mathy et al., *Impact of Simple Cheating in Application-Level Multicast*, In Proceedings of Infocom 2004, Hong Kong, March 2004. 98

[69] A. Walters, D. Zage and C. Nita-Rotaru, *Mitigating Attacks Against Measurement-Based Adaptation Mechanisms in Unstructured Multicast Overlay Networks*, In Proceedings of ICNP 2006, Santa Barbara, November 2006. 98

[70] R. Zhang, C. Tang, Y. C. Hu, S. Fahmy and X. Lin, *Impact of the Inaccuracy of Distance Prediction Algorithms on Internet Applications, an Analytical and Comparative Study*, In proceedings of Infocom 2006, Barcelona, April 2006. 98

2007.

[71] A. W. Moore et al., *Median Filtering for Removal of Low- Frequency Drift*, Analytic Chemistry, pp. 65:188, 1993. 100

[72] G. R. Shorack and J. A. Wellner, *Empirical Processes With Applications to Statistics*, John Wiley&Sons Inc, 1986, p.976. 102, 129

[73] M. de Berg, et al. *Computational Geometry, Algorithms and Applications*, Springer 2000. 107

[74] J. A. Rice, *Mathematical Statistics and Data Analysis*, Wadsworth & Brooks/Cole. Chapter 9. 1988. 128

Abstract

Internet coordinate-based systems allow easy network positioning. In such systems, the basic idea is that if network distances between Internet nodes can be embedded in an appropriate space, unmeasured distances can be estimated using a simple distance computation in that space. Recently, these coordinates-based systems have been shown to be accurate, with very low distance prediction error. However, most, if not all, of current proposals for coordinate systems assume that the nodes partaking in the system cooperate fully and honestly with each other – that is that the information reported by probed nodes is correct – this could also make them quite vulnerable to malicious attacks. In particular, insider attacks executed by (potentially colluding) legitimate users or nodes infiltrating the system could prove very effective.

As the use of overlays and applications relying on coordinates increases, one could imagine the release of worms and other malware, exploiting such cooperation, which could seriously disrupt the operations of these systems and therefore the virtual networks and applications relying on them for distance measurements.

In this thesis, we first identify such attacks, and through a simulation study, we observed their impact on two recently proposed positioning systems, namely Vivaldi and NPS. We experimented with attack strategies, carried out by malicious nodes that provide biased coordinates information and delay measurement probes, and that aim to (i) introduce disorder in the system, (ii) fool honest nodes to move far away from their correct positions and (iii) isolate particular target nodes in the system through collusion. Our findings confirm the susceptibility of the coordinate systems to such attacks.

Our major contribution is therefore a model for malicious behavior detection during coordinates embedding. We first show that the dynamics of a node, in a coordinate system without abnormal or malicious behavior, can be modeled by a Linear State Space model and tracked by a Kalman filter. Then we show, that the obtained model can be generalized in the sense that the parameters of a filter calibrated at a node can be used effectively to model and predict the dynamic behavior at another node, as long as the two nodes are not too far apart in the network. This leads to the proposal of a Surveyor infrastructure: Surveyor nodes are trusted, honest nodes that use each other exclusively to position themselves in the coordinate space, and are therefore immune to malicious behavior in the system. During their own coordinate embedding, other nodes can then use the filter parameters of a nearby Surveyor as a representation of normal, clean system behavior to detect and filter out abnormal or malicious activity. A combination of simulations and PlanetLab experiments are used to demonstrate the validity, generality, and effectiveness of the proposed approach for both Vivaldi and NPS.

Finally, we address the issue of asserting the accuracy of Internet coordinates advertised by nodes of Internet coordinate systems during distance estimations. Indeed, some nodes may even lie deliberately about their coordinates to mount various attacks against applications and

overlays.

Our proposed method consists in two steps: 1) establish the correctness of a node's claimed coordinate by using the Surveyor infrastructure and malicious embedding neighbor detection; and 2) issue a time limited validity certificate for each verified coordinate. Validity periods are computed based on an analysis of coordinate inter-shift times observed by Surveyors. By doing this, each surveyor can estimate the time until the next shift and thus, can limit the validity of the certificate it issues to regular nodes for their calculated coordinates. Our method is illustrated using a trace collected from a Vivaldi system deployed on PlanetLab, where inter-shift times are shown to follow long-tail distribution (log-normal distribution in most cases, or Weibull distribution otherwise). We show the effectiveness of our method by measuring the impact of a variety of attacks, experimented on PlanetLab, on distance estimates.

Résumé

Les systèmes Internet à base de coordonnées permettent un positionnement pratique des nœuds dans le réseau. Dans ce type de systèmes, l'idée principale est que si les distances réseau entre différents nœuds Internet peuvent être plongées dans un espace approprié, alors les distances non mesurées peuvent être estimées en utilisant une simple opération de calcul de distance géométrique dans cet espace. Récemment, on a pu prouver que ces systèmes à base de coordonnées étaient précis, avec une faible erreur de prédiction. Cependant, ces systèmes se basent souvent sur une coordination entre les nœuds, et font l'hypothèse que les informations reportées par les nœuds sont correctes.

Dans cette thèse, nous avons identifié plusieurs attaques, exploitant cette hypothèse d'honnêteté des nœuds, et pouvant être lancées contre des systèmes de positionnement Internet à base de coordonnées. Nous avons en l'occurrence étudié l'impact qu' avaient de telles attaques sur deux systèmes représentatifs des systèmes de positionnement actuels : NPS et Vivaldi. Nous avons entre autres montré que ces attaques, pouvaient dangereusement mettre en péril le bon fonctionnement de ces systèmes de coordonées, et par la même les applications se basant sur ce système pour les estimations de distances. À travers les simulations de plusieurs attaques, menées par des nœuds malhonnêtes, fournissant des coordonnées biaisées ou retardant les mesures, nous avons expérimenté plusieurs stratégies d'attaques qui ont pour objectifs: (i) d'introduire du désordre dans le système, (ii) de tromper les nœuds honnêtes afin qu'ils se positionnent loin de leurs coordonnées correctes et (iii) d'isoler certains nœuds cibles à travers des collusions. Nos résultas confirment la vulnérabilité de tels systèmes à ces attaques.

Notre contribution majeure a été par la suite de proposer un modèle de détection des comportements malicieux au sein de ces systèmes de positionnement durant le calcul des cordonnées.

Nous avons montré en premier lieu que la dynamique d'un nœud, dans un système de coordonnées, exempt de comportements anormaux ou malhonnêtes, peut être modélisée par un modèle d'états linéaire, et traqué par un filtre de Kalman. De plus, les paramètres d'un filtre calibré au niveau d'un nœud donné, peuvent être utilisés pour modéliser et prédire le comportement dynamique d'un autre nœud, tant que ces deux nœuds sont proches l'un de l'autre dans le réseau. Nous avons dès lors proposé une infrastructure de nœuds experts : des nœuds de confiance, se positionnant dans l'espace des coordonnées, en utilisant exclusivement d'autres nœuds experts. Ils sont alors immunisés contre n'importe quel comportement malicieux dans le système. Pendant le calcul de leurs propres coordonnées, les autres nœuds utilisent les paramètres du filtre d'un nœud expert proche, comme étant une représentation d'un comportement normal, pour détecter et filtrer toute activité malicieuse ou anormale. Une combinaison de simulations et d'expérimentations PlanetLab a été utilisée pour démontrer la validité, la généralité et l'efficacité de l'approche proposée pour chacun des deux systèmes Vivaldi et NPS.

Enfin, nous nous sommes penchés sur le problème de la validité des coordonnées Internet telles qu' annoncées par les nœuds d'un système de coordonnées durant la phase d'estimation des distances. En effet, certains nœuds peuvent délibérément mentir quant à la valeur exacte de leurs coordonnées afin de lancer diverses attaques contre les applications et les réseaux de couverture.

La méthode proposée se divise en deux étapes : 1)établir l'exactitude des coordonnées annoncées en utilisant l'infrastructure des nœuds experts et la méthode de détection des nœuds malicieux, et 2) délivrer un certificat à validité limitée pour chaque coordonnée vérifiée. Les périodes de validité sont calculées à partir d'une analyse des temps d'inter-changement observés par les nœuds experts. En faisant cela, chaque nœud expert, peut estimer le temps jusqu'au prochain changement de coordonnées, et ainsi, peut limiter le temps de validité du certificat qu'il délivrerait aux nœuds normaux. Notre méthode est illustrée en utilisant une trace recueillie à partir d'un système Vivaldi déployé sur PlanetLab, où les distributions de temps d'inter-changements suivent des distributions longue traine (distribution log-normale dans la plupart des cas, et distribution Weilbull sinon). Nous montrons l'efficacité de notre méthode en mesurant l'impact de plusieurs attaques sur les estimations de distance, expérimentées sur PlanetLab.