SECURING WIRELESS NETWORK TOPOLOGY AND ROUTING

by Weichao Wang

Center for Education and Research in
Information Assurance and Security,
Purdue University, West Lafayette, IN 47907-2086

SECURING WIRELESS NETWORK TOPOLOGY AND ROUTING

A Thesis

Submitted to the Faculty

of

Purdue University

by

Weichao Wang

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

August 2005

To my parents Kezheng Wang and Wenhuan Liu and my wife Aidong Lu.

ACKNOWLEDGMENTS

I would like to acknowledge the efforts of my major advisor, Professor Bharat Bhargava, in motivating and guiding me during my graduate study at Purdue. Professor Bhargava's expertise and insight are the keys for the success of his students. I have been learning from him not only the secret to conduct creative research, but also the principles as an investigator.

I would like to thank Professor Michael Zoltowski, Professor Mikhail Atallah, Professor Sonia Fahmy, and Professor Dongyan Xu for their valuable advices. The support and guidance from my advising committee have helped me overcome many difficulties.

I thank my colleagues in the RAID laboratory for their help and support. Dr. Leszek Lilien and Dr. Xiaoxin Wu gave me valuable comments on research. The discussions and collaboration with Dr. Yi Lu, Dr. Yuhui Zhong, and Mr. Gang Ding have been a tremendous experience for me.

This research is supported by NSF grants IIS-0209059 and CCR-0001788. I would also like to thank CERIAS and CISCO for the seeding fund.

Finally, I greatly appreciate the support and encouragement from my loving wife Aidong Lu.

TABLE OF CONTENTS

Page

LIST OF TABLES ix

LIST OF FIGURES x

ABSTRACT xii

1 Introduction 1

1.1 Problem statement 1

1.1.1 Protecting neighbor discovery in wireless networks 1

1.1.2 Protecting distance vector routing protocols in ad hoc networks . . 3

1.2 Thesis contributions 4

1.2.1 Defending against wormhole attacks in wireless networks 4

1.2.2 Identifying attackers in distance vector routing protocols 6

1.3 Thesis organization 6

2 Background 9

2.1 Neighbor discovery in wireless networks 9

2.2 Distance vector routing protocols for ad hoc networks 10

2.2.1 Introduction of DSDV 10

2.2.2 Introduction of AODV 10

2.3 Sensor networks 11

2.4 Simulation environment 12

3 COTA: Defending against wormhole attacks on ad hoc networks 13

3.1 Introduction 13

3.2 Problem statement 15

3.3 Related work 18

3.4 Assumptions and notations 20

3.4.1 Network assumptions 20

Page

3.4.2 Node assumptions 20
3.4.3 Key setup 22
3.4.4 Model of attackers 23
3.4.5 Notations 23
3.5 Detecting wormhole attacks 24
3.5.1 Basic end-to-end mechanism 24
3.5.2 Overhead of basic end-to-end mechanism 28
3.5.3 Cell-based open tunnel avoidance 30
3.5.4 Detection capability of COTA 32
3.5.5 Parameter determination 34
3.5.6 Data structure maintenance 38
3.5.7 Experiments on real devices 39
3.6 Controlling communication overhead 40
3.6.1 Frequency to conduct wormhole detection 40
3.6.2 Controlling byte overhead 42
3.7 Security analysis 42
3.8 Simulation and results 43
3.8.1 Simulation setup 44
3.8.2 Results on false alarms 45
3.8.3 Results on communication overhead 48
3.9 Discussions 50
3.10 Conclusion 51
4 Visualization of wormholes in sensor networks 53
4.1 Introduction 53
4.2 Related work 57
4.2.1 MDS and its applications in wireless networks 57
4.2.2 Wormhole detection in ad hoc networks 58
4.2.3 Underwater sensor networks 58

Page

4.2.4 Distance estimation between wireless nodes 58
4.2.5 Key distribution in sensor networks 59
4.3 MDS-VOW: A centralized approach 60
4.3.1 System assumptions 60
4.3.2 Network bootstrap 61
4.3.3 Building blocks of MDS-VOW 63
4.3.4 The MDS-VOW algorithm 69
4.4 Dis-VoW: A distributed approach 69
4.4.1 System assumptions 69
4.4.2 Building blocks of Dis-VoW 71
4.4.3 The Dis-VoW algorithm 79
4.5 Simulation results of MDS-VOW 79
4.5.1 Grid placement 81
4.5.2 Random placement 84
4.6 Simulation results of Dis-VoW 85
4.6.1 Grid placement 87
4.6.2 Random placement 89
4.7 Discussion 90
4.7.1 Impacts of sensor density 91
4.7.2 Safety of the proposed mechanisms 91
4.7.3 Control of false positive alarms 92
4.7.4 Frequency to conduct wormhole detection 92
4.8 Conclusion 93
5 Security study of ad hoc routing protocols 95
5.1 Introduction 95
5.2 Related work 96
5.3 Attack analysis and security comparison 97
5.3.1 Classification of attacks 98

Page

5.3.2 Security analysis 99

5.4 Simulation results 102

5.4.1 Simulation condition one 104

5.4.2 Simulation condition two 108

5.5 Detecting false destination sequence attacks 109

5.6 Conclusion 112

6 Defending against false destination sequence attacks in AODV 114

6.1 Introduction 114

6.2 Background 116

6.2.1 Evaluation criteria 116

6.2.2 Previous research on intruder identification 117

6.3 Reverse labeling restriction protocol 118

6.3.1 System assumption 118

6.3.2 Basic Ideas 119

6.3.3 Protocol details 121

6.4 Simulation results 124

6.4.1 Simulation environment 124

6.4.2 Effectiveness 126

6.4.3 Detection accuracy 127

6.4.4 Overhead 130

6.5 Discussion 132

6.6 Conclusion 133

7 Conclusions and future work 135

7.1 Conclusions 135

7.1.1 Wormhole detection in mobile ad hoc networks 135

7.1.2 Wormhole detection in sensor networks 136

7.1.3 Security study of distance vector routing protocols 137

7.1.4 Defending against false destination sequence attacks in AODV . . 138

Page

7.2 Future work 139
7.2.1 Effective representation of security data 139
7.2.2 Security in wireless mesh networks 140
7.2.3 Secure pervasive systems 140
LIST OF REFERENCES 141
VITA 150

LIST OF TABLES

Table		Page
3.1	Overhead comparison between the end-to-end mechanism and COTA	38
3.2	Computation efficiency of COTA	40
3.3	Simulation parameters	45
4.1	Distortions in edge lengths	76
5.1	Simulation parameters	103
6.1	Simulation parameters	125
6.2	Detection accuracy of RLR versus node pause time	128
6.3	Detection accuracy of RLR versus node pause time	128
6.4	Detection accuracy of RLR versus number of malicious nodes	129
6.5	Detection accuracy of RLR versus number of malicious nodes	129

LIST OF FIGURES

Figure Page

3.1 The classification of wormholes 17

3.2 Delivery of a wormhole detection packet in end-to-end mechanism 26

3.3 Cell division and data structure of COTA in a destination node 31

3.4 Storage space of COTA with different values of $sensitivity$ 37

3.5 False positive alarms: no wormholes, sensitivity / unit distance changes . . 46

3.6 False positive alarms: no wormholes, added offset / unit distance changes . 46

3.7 False positive alarms when wormholes exist in the network 47

3.8 False negative alarms when wormholes exist in the network 48

3.9 Packet overhead of the wormhole detection mechanism 49

3.10 Byte overhead of the wormhole detection mechanism 49

4.1 Wormhole attacks in underwater sensor networks 54

4.2 Impacts of wormholes on global reconstruction of 2D networks 55

4.3 Impacts of wormholes on global and local reconstruction of 3D networks 56

4.4 Impacts of distance measurement errors on network reconstruction 64

4.5 Smoothing the reconstructed network 65

4.6 The two-ended torch structure caused by a wormhole 67

4.7 Wormhole indicators of the sensors in different network scenarios 68

4.8 Generate B_i from the distance matrix D 74

4.9 Distortions in local reconstruction 75

4.10 $wormhole\ indicator$ values under different scenarios 78

4.11 Experimental network topology 80

4.12 Detection accuracy of MDS-VOW in grid placement when the error rate e_m changes 82

4.13 Increase in the average path length caused by false positive alarms 83

Figure Page

4.14 Detection accuracy of MDS-VOW when varying wormhole length 83
4.15 Detection accuracy of MDS-VOW in random placement 84
4.16 Experimental network topology 86
4.17 Detection accuracy of Dis-VoW with different error rate e_m 87
4.18 Detection accuracy of Dis-VoW with different numbers of wormholes . . 88
4.19 Detection accuracy of Dis-VoW in random placement of sensors 90
5.1 The delivery ratio versus the number of connections 105
5.2 The number of cheated nodes versus the number of connections 106
5.3 Number of false route packet versus the number of connections 106
5.4 The number of cheated nodes versus the false route packet 107
5.5 The delivery ratio versus the mobility of nodes 108
5.6 The attack overhead versus the mobility of nodes 109
5.7 The anomalous patterns of sequence number in DSDV 110
5.8 The anomalous patterns of sequence number in AODV 111
6.1 Routing forest of a mobile node with false routing trees 120
6.2 Data packet delivery ratio versus node pause time 126
6.3 Data packet delivery ratio versus the number of independent attackers . . 127
6.4 Normalized control packet overhead versus node pause time 130
6.5 Computation overhead of RLR versus node pause time 131
6.6 Computation overhead of RLR versus the number of attackers 131

ABSTRACT

Wang, Weichao. Ph.D., Purdue University, August, 2005. Securing Wireless Network Topology and Routing. Major Professor: Bharat Bhargava.

This dissertation investigates two research problems to protect wireless network topology and routing: (1) designing protocols with configurable overhead to defend against wormhole attacks; (2) designing an intruder identification mechanism to locate and isolate the malicious nodes in distance vector routing protocols.

Previous approaches for wormhole detection in ad hoc networks assume a relationship of trust between direct neighbors and cannot detect wormholes when the attackers are legal members in the network. As a generic approach, an end-to-end mechanism is proposed that assumes trust only between the source and the destination of a route. It integrates the positions of nodes and loosely synchronized clocks to identify fake neighbor connections. An information management scheme is designed to allow a mobile node to predetermine the resources that are consumed on wormhole detection. In our experiments, the computation overhead is less than 0.28% of the CPU time for a ten-hop route. This justifies the feasibility of the proposed mechanism.

For wormhole detection in sensor networks, we propose the first group of approaches that do not depend on any special hardware. A normalized variable $wormhole\ indicator$ is defined based on the distortions in edge length and angles among neighboring sensors. As a centralized approach, MDS-VOW reconstructs the network layout using inaccurate distance measurements among sensors and identifies fake neighbor connections. As a distributed approach, Dis-VoW allows every sensor to detect wormholes locally when the network topology changes. The research creates a new method to solve wireless network security problems by integrating techniques from social science, computer graphics, and scientific visualization.

An intruder identification mechanism is designed to locate and isolate malicious nodes that attack the AODV protocol with false destination sequence numbers. The propagation paths of false routes are marked through reverse labeling and the suspicious attackers are put into blacklists to achieve isolation. The quorum voting method is adopted to reduce false positive alarms. In our experiments, the proposed mechanism can improve the packet delivery ratio by 30% even when there are multiple malicious nodes in the network.

1 INTRODUCTION

1.1 Problem statement

The research problem is *how to provide security protection to the network topology and the routing procedure in a wireless network.* The major challenges include dynamic topology, decentralized control, limited resources, and the lack of information dissemination control. We investigate the problem in two network environments: (1) mobile ad hoc networks; (2) sensor networks. As building blocks for securing wireless network topology and routing, two sub-problems are of special interest.

1.1.1 Protecting neighbor discovery in wireless networks

Neighbor discovery procedures play an important role in wireless networks. The current neighbors usually construct the set of candidates for next hop during the packet forwarding. There have been approaches using neighbors to conduct distributed node behavior monitoring to improve security in wireless networks. Therefore, mechanisms must be designed to protect such procedures.

Although the adoption of encryption and authentication methods can prevent a malicious node from impersonating another node during neighbor discovery, there are attacks which cannot be prevented by such schemes. For example, since the wireless nodes use a radio channel to send information, the attackers can eavesdrop the packets, tunnel them to another location in the network, and retransmit them. This generates a false scenario that the original sender is in the neighborhood of the remote location. The tunneling procedure forms a wormhole and it can be conducted by collusive, weak attackers without compromising the secret keys.

Wormhole attacks put severe threats to both routing protocols and some security enhancements. In many routing protocols, wireless nodes depend on the neighbor discovery procedure to construct the local network topology. If the attackers tunnel the neighbor discovery beacons through wormholes, the good nodes will get false information about their neighbors. This may lead to the choice of a non-existent route.

When the differences in processing capability and available resources in mobile ad hoc networks and sensor networks are considered, different mechanisms must be designed to defend against wormhole attacks in these environments. In ad hoc networks, an end-to-end mechanism should be developed so that it can detect wormholes along a multi-hop path even when the relationship of trust between the intermediate nodes cannot be assumed. In sensor networks, the proposed mechanism should not depend on any special hardware such as synchronized clocks, positioning devices, or directional antenna so that the deployment of the proposed mechanism will not increase the hardware cost. We plan to study the following research problems:

- What are the different formats of the wormholes? How does the format affect the detection capability of the mechanisms?
- How to detect wormhole attacks in ad hoc networks when only the source and the destination of a route trust each other?
- How to manage the detection information efficiently without degrading the detection capability?
- How to detect wormholes in a sensor network when only imprecise distance measurements between neighbors are available? How to design the mechanism that is robust to distance measurement errors?

The research will result in the following contributions:

- A classification of wormholes will establish a basis on which the detection capabilities of various approaches can be identified.

- Development of the mechanisms that can detect wormholes in ad hoc networks and sensor networks. Design of a detection information management scheme whose storage and computation overhead can be controlled by the wireless nodes.
- A better understanding of the impacts of wormhole attacks on the topology of wireless networks will be provided. The results will lead to the design of new mechanisms to protect the network topology in resource restraint wireless environments.

1.1.2 Protecting distance vector routing protocols in ad hoc networks

In a self-organized environment such as an ad hoc network, the wireless nodes are deployed in a method that they help each other to establish routes and forward packets. Since there does not exist a pre-defined architecture, the network is vulnerable to the attacks of false routes. The malicious nodes can attract the traffic by providing a fake route and then drop or manipulate the data packets. With the wide adoption of distance vector technique in the routing protocols, the fake route can appear to be shorter in hops, shorter in delay, or with a fresher sequence number.

Being aware of the vulnerabilities, the researchers have presented various intrusion detection mechanisms and security enhancements to the routing protocols. The proposed research will investigate the connections between the characteristics of the routing protocols and their vulnerabilities. The research will lead to an intruder identification mechanism that can locate and isolate the malicious nodes. Network performance will be improved by isolating the suspicious attackers. We plan to investigate the following research problems:

- Study the vulnerabilities of and attacks on the routing protocols, especially the difference caused by on-demand and proactive route queries.
- Investigate the efficiency and accuracy of the reverse labeling method for intruder identification in distance vector routing protocols.
- Conduct experimental studies to investigate the detection accuracy, overhead, and performance improvement brought by the proposed mechanism.

The research will result in the following contributions:

- Provide a better understanding of the relationship between the properties of the ad hoc routing protocols and their security vulnerabilities.
- Establish the evaluation criteria for the intruder identification mechanisms which can be applied as the basis for future performance comparison.
- Propose an intruder identification protocol that can be integrated with distance vector routing protocols for ad hoc networks to improve security.

1.2 Thesis contributions

1.2.1 Defending against wormhole attacks in wireless networks

Wormhole attacks threaten the safety of ad hoc networks. We investigate the detection capability of previous approaches and find that they can only detect a subset of the wormholes because they assume a relationship of trust between the neighboring nodes. As a more generic approach, an end-to-end mechanism that can detect wormholes along a multi-hop route is proposed. Only trust between the source and the destination is assumed. The mechanism combines geographic information and loosely synchronized clocks to detect anomalies in neighbor relations and node movements. To reduce the computation and storage overhead, we present a scheme, Cell-based Open Tunnel Avoidance(COTA), to manage the information. COTA achieves a constant space for every node on the path and the computation overhead increases linearly to the number of detection packets. We prove that the savings do not deteriorate the detection capability. With COTA, the mobile node can predetermine the resources that it wants to consume on wormhole detection and control the tradeoff between the achieved security level and the overhead. The contributions can be summarized as:

- A classification of wormholes that can be adopted to identify the detection capability of various approaches.

- An end-to-end wormhole detection mechanism that assumes the trust relationship only between the source and the destination. It can detect wormholes which cannot be discovered by previous solutions when the malicious nodes are “insiders”.

- A configurable position and movement information management scheme for wireless networks.

We propose two mechanisms, Multi-Dimensional Scaling-Visualization of Wormhole (MDS-VOW) and Distributed Visualization of Wormhole (Dis-VoW), to defend against such attacks in sensor networks. Both schemes do not require the sensors to be equipped with any special hardware. As a centralized approach, MDS-VOW first reconstructs the layout of the sensors using multi-dimensional scaling based on the inaccurate distance estimations between the neighboring nodes. To compensate the distortions caused by distance measurement errors, a surface smoothing scheme is adopted. MDS-VOW then detects the wormhole by visualizing the anomalies introduced by the attack. The anomalies, which are caused by the fake connections through the wormhole, bend the reconstructed surface to pull the sensors that are faraway to each other. Through detecting the bending feature, the wormhole is located and the fake connections are identified.

Dis-VoW is a distributed approach. It allows the sensors to reconstruct the local network and use the distortions in edge length and angles among the neighboring sensors to locate the fake neighbor connections. Less computation and storage is consumed and distributed detection can be conducted when the network topology changes. The contributions can be summarized as:

- MDS-VOW and Dis-VoW are the first wormhole detection mechanisms that do not require the sensors to be equipped with any special hardware.

- Both MDS-VOW and Dis-VoW integrate the techniques from social science, computer graphics, and scientific visualization to attack network security problems. This will help to promote the inter-disciplinary research in wireless security.

- As a distributed approach, Dis-VoW can conduct localized wormhole detection when the network topology changes because of sensor movement.

1.2.2 Identifying attackers in distance vector routing protocols

Analysis and extended simulations have been conducted to study the vulnerabilities of Ad hoc On-demand Distance Vector protocol (AODV) and Destination-Sequenced Distance Vector protocol (DSDV) for ad hoc networks. To defend against the false destination sequence attacks, we propose a Reverse Labeling Restriction (RLR) mechanism to detect the false routing information and identify the malicious nodes. The propagation trees of the false routes are reconstructed using reverse labeling and the trees have a high probability to converge at the malicious nodes. Through ignoring the routing information provided by the attackers, we isolate the malicious nodes and significantly improve the network performance. The detection accuracy and overhead of RLR are investigated through simulation. The contributions can be summarized as:

- Vulnerability analysis and security comparison between the on-demand and proactive properties of the routing protocols for ad hoc networks.
- The development and evaluation of an intruder identification protocol RLR that can defend against false destination sequence attacks by identifying and isolating malicious nodes.

1.3 Thesis organization

Chapter 2 provides the background information for the research in this dissertation. It briefly introduces the neighbor discovery procedure in wireless networks and its vulnerabilities. Two distance vector routing protocols, DSDV and AODV, for ad hoc networks are described. An overview of sensor networks is presented and we focus on the features such as the limited available resources. The network simulator ns2 and the extended module to investigate various attacks on wireless networks are described.

Chapter 3 investigates wormhole detection in mobile ad hoc networks. A classification of the attacks based on the format of the wormholes is proposed. The analysis shows that earlier approaches focus on the prevention of wormholes among neighbors that trust each other. As a more generic approach, we present an end-to-end scheme that can detect wormholes on a multi-hop route. Only the trust between the source and the destination is assumed. To reduce the computation and storage overhead, we present a scheme called cell-based open tunnel avoidance (COTA) to manage the information. We prove that the savings do not deteriorate the detection capability. The schemes to control communication overhead are studied. The simulation and experiments on real devices show that the proposed mechanism can be integrated with existent routing protocols in ad hoc networks.

Chapter 4 investigates wormhole detection in sensor networks. We first illustrate how the wormholes will impact the network reconstruction using multi-dimensional scaling. As a centralized approach, we introduce the ideas of MDS-VOW and describe its building blocks in detail. The simulation results show that MDS-VOW has a low false alarm ratio when the distance measurement errors are not large.

We study the wormhole detection in underwater sensor networks. Two features, the slow signal propagation speed and node movement, are of special interest. A distributed approach Dis-VoW is developed that can detect wormholes by visualizing the distortions in the reconstructed local networks. Scenarios with different numbers of wormholes are simulated and the results show that Dis-VoW can detect most of the fake connections without introducing many false positive mistakes.

Chapter 5 compares the security properties of the AODV and DSDV protocols, especially the differences caused by on-demand and proactive route queries, by exploiting the attacks on the protocols. It investigates the damages of false distance vector attacks and false destination sequence attacks through simulation. The anomalous patterns of sequence numbers that can be used to detect such attacks are then presented.

Chapter 6 investigates intruder identification in the AODV protocol. The set of criteria to evaluate the algorithms are established. Reverse Labeling Restriction (RLR) is proposed to identify and isolate the malicious nodes that attack AODV with false destination sequence numbers. The simulation of RLR shows that most of the innocent nodes can identify all attackers. Isolating the malicious nodes brings a sharp increase in the packet delivery ratio.

Chapter 7 concludes the dissertation and describes the future research directions.

2 BACKGROUND

2.1 Neighbor discovery in wireless networks

In a wireless network, the nodes may roam over a wide area that is much larger than the radio coverage range of a single node. At the same time, the nodes may dynamically leave the system because of the hardware problems or the exhaustion of power. Therefore, the set of neighbors with which a wireless node can directly communicate will change constantly. The neighbor discovery procedure in a wireless network will help a mobile node to maintain the list of its current radio neighbors. It provides the real time view of the local network topology and the candidates of next hop during packet forwarding.

The beacon assisted neighbor discovery method has been adopted by many protocols. In this scheme, a wireless node will periodically broadcast a beacon or a HELLO message on a selected channel. The packet provides the information such as the node type and the address. The other nodes will scan this channel and update their neighbor lists based on the received beacons. A node will be removed from the neighbor list if the local host has not received the beacons from it for a pre-determined period of time.

Various approaches have been proposed to improve the performance of the neighbor discovery method in different environments. The adopted methods include using directional antennas [1], integrating it with MAC protocols [2], and improving energy efficiency [3].

Since the discovered neighbors play an important role in the routing protocols and the packet forwarding, security methods must be designed to protect the authenticity and integrity of the beacons. For example, authentication methods can be adopted to verify the identity of the sender. There are some attacks, such as the wormhole attack, in which the malicious nodes only eavesdrop and retransmit the beacons to confuse the neighbor discovery. In Chapter 3 and 4 we will study their impacts and design mechanisms to defend against such attacks.

2.2 Distance vector routing protocols for ad hoc networks

2.2.1 Introduction of DSDV

DSDV is based on distance vector technology. Every node broadcasts its routing table routinely, which enables the proactive route discovery. When one link change that may severely impact the connectivity happens, a partial update can also be sent. To avoid the routing loop, DSDV uses the destination sequence number to identify the freshness of the routing information. The sequence number of a specific node is increased at every time when it sends out the route update. The route with the largest sequence number is always preferred. When multiple paths with the same sequence are available, the shortest one will be selected. More details about DSDV can be found in [4].

DSDV's desirable feature is the short delay of connections brought by proactive route discovery. Since every node has to maintain a routing table that covers all nodes, DSDV does not scale well to large networks. The overhead of recalculation of routes and periodical packet exchanges consume the valuable resources of energy and bandwidth. However, the proactive mechanism sets up difficulties for the malicious nodes to conduct attacks. This will be shown by the analysis and simulation results in Chapter 5.

2.2.2 Introduction of AODV

AODV is a reactive protocol and it is based on distance vector technology. When the source node wants to send packets to the destination and cannot get the route from its routing table, it will broadcast a Route Request (RREQ). The receivers may establish the routes back to the source through the RREQ. If the receiver has an active route to the destination, it will unicast a Route Reply (RREP). Otherwise, the RREQ will be rebroadcast further. If a reply is sent, all nodes along that path may establish the route to the destination. To prevent the same request from being broadcast repeatedly, every request is identified by a <Node ID, Broadcast ID> couple. The mobile nodes send out the Route Error (RERR) packets to report broken paths.

To avoid routing loop and to identify the freshness of the route, destination sequence number is introduced. The sequence of a mobile node is increased at every time that it sends RREQ or RREP. The sequence in RREP must be larger than or equal to the one carried in corresponding RREQ. The path with the largest destination sequence number is always preferred. If several paths have the same sequence, the shortest one will be chosen. More details about AODV can be found in [5].

AODV's desirable features are its low byte overhead and loop free routing. But the on-demand route query usually brings a longer delay for the first few packets. The genuineness of the destination sequence and distance vector leaves vulnerabilities to attackers. These deficiencies introduce the attacks that will compromise the network.

2.3 Sensor networks

A wireless sensor network consists of a large number of low cost, small, and smart computing devices. The sensors usually act autonomously and communicate with each other through a short range radio channel. Both static and movable sensor networks have been deployed to achieve distributed monitoring of various environments, real time information collection and integration, and the exploitation of unknown areas.

A sensor usually has weak processing capability, limited storage space, narrow bandwidth, and restricted power. Therefore, the research has been focusing on how to use these resources efficiently and effectively. Since the processing capability of an individual sensor is relatively weak, our research in sensor networks focuses on designing new protocols to enforce security through integrating the information from the large number of entities. The proposed mechanisms do not depend on any special hardware.

2.4 Simulation environment

We have conducted extended simulation using ns2 [6] during our investigation to study the performance and robustness of the proposed mechanisms. ns2 is an event driven network simulator that has been widely adopted by the network researchers. In our simulation, the wireless nodes share a radio channel and a method similar to the IEEE 802.11 distributed coordination function is adopted as the MAC layer protocol.

We have extended ns2 with a new module and a set of interfaces that can simulate both internal and external attacks on the network. The attackers communicate with the innocent nodes through the public channel. They are equipped with a dedicated channel that does not interfere with the public channel and can be used to conduct wormhole attacks. The malicious node class now supports AODV and DSDV routing protocols. The implemented attacks include false destination sequence number, false distance vector, malicious broadcast, and wormhole attacks.

3 COTA: DEFENDING AGAINST WORMHOLE ATTACKS ON AD HOC NETWORKS

3.1 Introduction

As ad hoc networks are merging into the pervasive computing environment, security becomes a central requirement. Distributed node behavior monitoring has been applied to enhance security. A system integrating watchdog and pathrater with the Dynamic Source Routing protocol (DSR) [7] is presented in [8]. In security enhanced Ad hoc On-demand Distance Vector protocol (AODV-S) [9], the neighbors collaboratively authorize a token to the node before it joins the network activities. The researchers have proposed several protocols that use hash chains or digital signatures to protect the integrity and authenticity of routing information. The Secure AODV protocol (SAODV) [10] adopts both mechanisms. Secure Efficient Ad hoc Distance vector routing (SEAD) [11] and Ariadne [12] use a variant of the Timed Efficient Stream Loss-tolerant Authentication (TESLA) [13] to accomplish authentication. A security-aware routing environment has been presented in [14].

Intrusion detection systems (IDS) have been adopted as the second line of defense to protect ad hoc networks. Zhang and Lee presented a generic multi-layer integrated IDS structure [15]. Bharghavan [16] and Haas *et al* [17] explored the security issues in wireless LANs and ad hoc networks respectively. An architecture that combines IDS with trust is presented by Alberts *et al* [18]. Intrusion detection using mobile agents is studied in [19].

In this chapter, we focus on the detection of wormhole attacks in ad hoc networks. Since the mobile devices use a radio channel to send information, the malicious nodes can eavesdrop the packets, tunnel them to another location in the network, and retransmit them. This generates a false scenario that the original sender is in the neighborhood of the remote location. The tunneling procedure forms a wormhole. It is conducted by collusive attackers. If a fast transmission path (e.g. a dedicated channel shared by attackers) exists between the two ends of the wormhole, the tunneled packets can propagate faster than those through a normal multi-hop route. This forms the "rushing attack" studied by Hu *et al* [20].

Wormhole attacks put severe threats to both ad hoc routing protocols and some security enhancements. In many routing protocols, mobile nodes depend on the neighbor discovery procedure to construct the local network topology. If the attackers tunnel the neighbor discovery beacons through wormholes, the good nodes will get false information about their neighbors. This may lead to the choice of a non-existent route. Zero-interaction authentication (ZIA) [21] is designed to protect the data on mobile devices from the illegal access. The files are decrypted only when an authentication token that is worn by the user can directly communicate to the device through a short-range wireless channel. If a wormhole exists between the token and the device, the data may be disclosed.

Some efforts have been put on this problem and encouraging results have been collected [22–24]. However, the classification in Section 3.2 shows that the previous research focuses on the detection of closed wormholes. Half open and open wormholes, whose detection requires information exchanges beyond direct neighbors, have not been studied. In this chapter, we propose an end-to-end mechanism that can detect wormholes along a multi-hop path on which the intermediate nodes do not necessarily trust each other. The mechanism combines authentication with location information. To prevent the destination node from being overwhelmed by the computation and storage overhead, a scheme called Cell-based Open Tunnel Avoidance (COTA) is introduced to manage the position records. COTA enables a node to pre-determine the resources that it wants to consume on wormhole detection. It reduces the overhead without hurting the detection capability. The frequency to conduct the detection is also studied to examine the communication overhead. The proposed mechanism can be combined with existent ad hoc routing protocols.

The contributions of this chapter are:

- A classification of the wormholes is proposed. We divide the attacks into three groups (closed, half open, and open) according to the format of the tunnel and attacker's capability. The classification establishes a basis on which the detection capability of the approaches can be identified.

- We propose an end-to-end mechanism that can detect closed, half open, and open wormholes in ad hoc networks. Considering the limited resources available to a mobile node, we design COTA to manage the information. The communication overhead introduced by the proposed mechanism is also studied.

- Simulation is conducted to evaluate the overhead, detection capability, and accuracy of the proposed mechanism. Experiments on off-the-shelf mobile devices are conducted to examine the practicability of this method in real world.

The remainder of this chapter is organized as follows: Section 3.2 describes the wormhole attacks and their impacts on ad hoc network security. The classification is given out. In Section 3.3, we review the previous work. Section 3.4 presents and justifies our assumptions. Section 3.5 describes the end-to-end mechanism and COTA in detail. The problems such as detection capability and parameter determination are studied. Section 3.6 studies the frequency to conduct wormhole detection and the method to control communication overhead. Section 3.7 investigates the robustness of the proposed mechanism. Section 3.8 presents the simulation results. Section 3.9 discusses the future work and Section 3.10 concludes the chapter.

3.2 Problem statement

In a wormhole attack, if the malicious nodes have a dedicated channel, the tunneling procedure can be conducted in real time. Since the packets are resent in the exactly same way, encryption or authentication alone cannot prevent the attacks. Other nodes cannot tell whether the packets are from the real originator or from the resender. A group of collusive attackers can form a wormhole that has as many ends as the number of malicious nodes.

Wormhole attacks put severe threats to ad hoc routing protocols. In the protocols that use distance vector technique, such as Ad hoc On-demand Distance Vector protocol

(AODV) [5] and Destination-Sequenced Distance Vector protocol (DSDV) [4], the hop count of a path affects the choice of routes. A pair of attackers can form a long tunnel and fabricate the false scenario that a short path exists between the source and the destination. The fake path will attract the data traffic. As soon as the packets are absorbed to the wormhole, the attackers can either drop them or compromise them. The attacks can also do harm to the hierarchical routing protocols such as Hierarchical State Routing protocol (HSR) [25], Clusterhead Gateway Switch Routing protocol (CGSR) [26], and Adaptive Routing using Clusters (ARC) [27]. They may confuse the clustering procedure and lead to a wrong topology. If a wormhole controls the link between two clusterheads or a link close to the root of the routing hierarchy, it can partition the network.

The safety and effectiveness of some security enhancements for ad hoc networks would be improved if wormholes can be defended. The example on ZIA has been shown in Section 3.1. Another example shows the impacts of such attacks on the distributed monitoring of node misbehaviors. In AODV-S [9], the neighbors collaboratively authorize a token to the node before it joins the network activities. If a wormhole exists beside the misbehaved node, the attackers can selectively tunnel the good-looking packets to the remote side. The good nodes at the remote side monitor all these packets and can not detect any security violations. The new token will be authorized. This may conflict with the conclusion drawn by the real neighbors. We can settle this embarrassment through preventing wormholes.

The classification of such attacks will facilitate the design of detection methods. According to whether the attackers are visible on the route, we classify the wormholes into three types: closed, half open, and open. The examples that include two malicious nodes are shown in Figure 3.1. For the clearance of the remainder of the chapter, we use $M1$ and $M2$ to represent the malicious nodes, S and D to represent the good nodes as source and destination, and A, B, $etc.$ as the good nodes on the route.

The nodes between the braces are those on the path but invisible to S and D because they are in a wormhole. We borrow the words “closed” and “open” from the definition of an interval, where “closed” means “start from and include”, and “open” means “start from

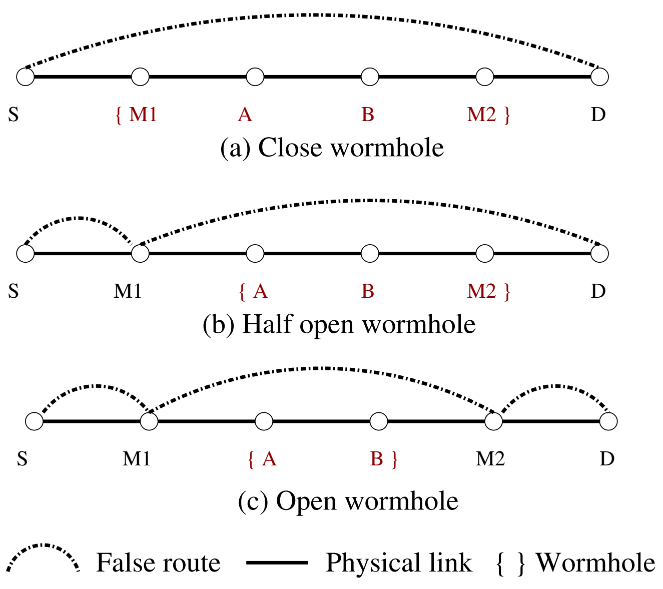


Figure 3.1. The classification of wormholes

but not include". In Figure 3.1a, $M1$ and $M2$ tunnel the neighbor discovery beacons from S to D, and vice versa, so S and D think that they are direct neighbors. Both $M1$ and $M2$ are in the wormhole. In Figure 3.1b, $M1$ is a neighbor of S and it tunnels its beacons through $M2$ to D. Only one malicious node is visible to S and D. In an open wormhole, both attackers are visible to S and D, as shown in Figure 3.1c.

The previous research focuses on the prevention of closed wormholes. The mechanisms that only examine direct neighbors cannot guarantee the detection of the other two types. For example, S and $M1$, and D and $M2$ are real neighbors in Figure 3.1c. This does not impact the establishment of an open wormhole between $M1$ and $M2$. Therefore, an end-to-end mechanism must be designed to defend against the half open and open wormholes.

Wormhole detection needs to be conducted when a neighbor relation or a route is first established. Besides, it must be conducted repeatedly during the lifetime of the neighbor relation or the route because the nodes are moving and wormholes can be formed dynamically. The detection frequency impacts the overhead and the detection accuracy. This problem is studied in Section 3.6.

3.3 Related work

The wormhole attackers encapsulate the received packets into their packets and tunnel them to another location. Ironically, this technique was first introduced to overcome some difficulties in networking. IP-tunnel is used in Mobile IP to transfer packets from home agents to remote agents [28]. In Virtual Private Networks (VPN), the tunnels are used to connect two LANs, allowing them to share data, but keeping them behind the firewalls [29]. IP-tunnels are also used in Multi-Protocol Label Switch (MPLS) [30].

Wormhole attacks threaten the safety of Internet routing protocols such as RIP2 [31]. But the attacker needs to have a total control on a router, which is not easy to achieve. Furthermore, through using static routes, such attacks can be prevented. Because of the dynamic membership and frequent topology changes, a node in ad hoc networks does not have this luxury.

Wormhole attacks on mobile ad hoc networks were independently discovered by Dahill *et al* [32], Hass *et al* [33], and Hu *et al* [22]. To defend against them, some efforts have been put on hardware design and signal processing techniques. If the data bits are transferred in some special modulating method known only to the neighboring nodes, they are resistant to the closed wormholes. Another approach, RF watermarking, works in the similar way. It modulates the radio waveform in a specific pattern to accomplish authentication. Both mechanisms will be compromised if the malicious nodes can accurately capture the signal patterns. Neither of them can prevent half open or open wormholes.

The adoption of directional antenna [34, 35] by mobile devices can raise the security levels. A solution that uses such equipments to defend against closed wormholes has been presented in [24]. The nodes examine the directions of the received signals from each other and a witness. Only when the directions of both pairs match, the neighbor relation is confirmed.

Another potential solution is to integrate the prevention methods into Intrusion Detection Systems. The traffic monitoring module of IDS will find that the ends of wormholes act as packet sinks: many data packets which are not destined to them will lose their tracks at these nodes. The joint response generated by the neighbors of the malicious node will expose the anomalous traffic pattern.

Some mechanisms proposed to locate the position of a mobile node in an indoor environment [36–38] can be applied to prevent wormholes. For example, both the original packet and the resent one will be captured by the location sensors and two conflicting positions of the same node will be detected. Either the good nodes or a centralized controller will discover this anomalous result. However, it will not be easy to port such methods to outdoor environments.

One approach to detect closed wormholes without clock synchronization is proposed by Capkun $et\ al$ in [23]. Every node is assumed to be equipped with a special hardware that can respond to a one-bit challenge without any delay. The challenger measures the round trip time of the signal with an accurate clock to calculate the distance between the nodes. The probability that an attacker can guess all bits correctly decreases exponentially as the number of challenges increases.

Another approach to detect closed wormholes is packet leash, which was proposed by Hu, Perrig and Johnson [22]. The leash is the information added into a packet to restrict its transmission distance. In the geographical leashes, the location information and loosely synchronized clocks together verify the neighbor relation. In temporal leashes, the packet transmission distance is calculated as the product of signal propagation time and the speed of light. Both mechanisms use lightweight hash chains to authenticate the nodes [13]. The Message Authentication Code (MAC) can be calculated in real time.

One advantage of packet leashes is the low computation overhead. Similar to geographical leashes, the end-to-end mechanism proposed in this chapter assumes the knowledge of location information and loosely synchronized clocks. It can be deployed in the environments where geographical leashes can be used to detect closed, half open, and open wormholes.

3.4 Assumptions and notations

3.4.1 Network assumptions

We assume that the links among nodes are bidirectional. Two neighboring nodes can always send packets to each other. This assumption will hold under most conditions in the real environment. Many medium access control protocols [39, 40] are also based on this assumption.

The security threats to mobile ad hoc networks come from all layers. The malicious node can jam the physical layer. There have been approaches using spread spectrum [41] to provide resistance to such attacks. There are also Deny of Service (DoS) attacks on the medium access control layer [42]. For example, if a malicious node keeps sending noises and causes collisions, the communication within the neighborhood will be paralyzed. The fairness control mechanisms such as time division multiple access [43] can avoid one attacker eating up all available bandwidth. In this chapter we will not discuss solutions to those attacks.

The network itself acts in Byzantine manners [44]. It may drop or corrupt packets. The packets can be duplicated or forwarded out of order.

3.4.2 Node assumptions

A node can locate its geographic position. There have been off-the-shelf devices, such as Global Positioning System (GPS) [45], that can provide accurate positioning service. The accuracy of position information will affect the detection capability of the proposed mechanism. For example, if the direct communication range among neighbors is d and the maximum error of the distance between two locations is δ, they introduce a relative error up to δ/d. The mechanisms such as GPS can locate the position with the accuracy of 15 feet.

In certain environments, using the distance between two nodes to verify the neighbor relation cannot guarantee the detection. For example, even if an obstacle (e.g. a building) exists between two nodes and blocks all signals, a position based mechanism may still consider them as neighbors. To detect such conditions, the nodes need a more accurate signal propagation model in that environment. It is able to determine whether the two nodes can actually communicate to each other when the positions are given. In this chapter, we assume that any two nodes having a distance shorter than d can communicate to each other.

Each node has a clock, and we assume that the clocks are loosely synchronized with the maximum relative error Δ. By this we mean that the difference between the clocks of any two nodes is smaller than Δ if we sample the time at the same moment.

The mobile nodes have limited computation resources. But they can accomplish the operations required by the security mechanisms, such as the calculation and verification of digital signatures, encryption using symmetric keys, and calculation of the MAC code.

We do not assume that a node can control the longest buffering time of a packet before it is forwarded. We do not assume trusted hardware such as tamper-proof wireless cards. They may restrict the deployment of the proposed mechanism.

Assumption justification

A lot of research in ad hoc networks has been conducted based on the position awareness assumption. They cover from routing [46–48], energy consumption [49, 50], to location based services [51, 52]. Some work has been conducted based on loose synchronization assumption, such as geographical leashes [22], Secure Tracking of Node Encounters (SECTOR) [23], Ariadne [12], and Light-weight Hop-by-hop Authentication Protocol (LHAP) [53]. In this part, we justify our assumptions based on the available techniques and application scenarios.

Location-based services using GPS are ready to launch with the progresses that reduce expense, size, weight, and power consumption of such devices. Motorola has unveiled one kind of GPS chip set whose unit cost is about $10 [54]. It is cheap and small enough to fit in PDAs and cell phones. Sprint PCS alone has sold over 8.8 million GPS-enabled handsets by March, 2003 [55]. The positioning device is becoming a common part of the wireless node for both military and civilian usage.

A lot of efforts have been put on clock synchronization in ad hoc and sensor networks. There have been solutions based on LORAN-C [56], WWVB [57], Reference Broadcast [58], TINY/MINI-SYNC [59], Tree-based solution [60], and GPS [61]. A good survey can be found in [60]. In this chapter we only assume loosely synchronized clocks among nodes, for example, $\Delta \leq 1 second$. If every node is equipped with GPS, its clock is accurate enough to satisfy this requirement. The mobile nodes do not need extra hardware or a complex protocol to get more accurate time.

The security of GPS has been studied for some time. Solutions for anti-jamming and anti-spoofing civilian GPS signals have been proposed in [62–64]. Here we do not present the details. GPS signals can be weak or biased in an indoor environment. But as discussed in Section 3.3, a more accurate positioning scheme in such an environment can be implemented to defend against wormholes.

3.4.3 Key setup

The safety of the end-to-end mechanism relies on the secrecy and authenticity of the keys stored in nodes. The authentication can be accomplished by pairwise secret keys, digital signatures, or a variant of TESLA [13].

The advantage of pairwise keys is that the nodes can use symmetric cryptographic methods and avoid the expensive operations such as exponential computation. It is very important to the mobile nodes with limited resources. The disadvantages are two folds. First, if there are n nodes in the network, we need to set up $n(n-1)/2$ keys. This can impact the scalability of the proposed mechanism. Second, it will be difficult to authenticate multicast or broadcast packets. Although there has been research on allowing multiple users to share a group key, it will put threats to the whole scheme if one node is compromised [65].

If digital signatures are used, we only need to set up n keys. The method supports the authentication of multicast traffic and has the non-repudiation property. But the computation of a digital signature can be three orders of magnitude slower than that of the symmetric mechanisms [66]. With the emergence of high speed signature algorithms [67, 68] and more powerful portable processors, computing digital signatures will cause less overhead on the mobile nodes.

With the assumption of loosely synchronized clocks, if the end-to-end delay can be accurately predicted, a variant of TESLA can be applied to accomplish authentication. We only need to set up n keys and the nodes do not have to do the expensive computations. The disadvantage is that the packets may not be authenticated immediately. The receiver needs to temporarily buffer the packet until the key is disclosed. This may require more storage space.

To set up the keys, either centralized mechanisms, such as a key distribution center or a Public Key Infrastructure (PKI), or a pre-load method during initialization, can be applied. A survey of key establishment in mobile networks can be found in [69]. Another solution proposed in [70] applies a PGP-like mechanism to distribute the keys.

3.4.4 Model of attackers

The attackers do not have the computation power to crack the secret keys. To tunnel the packets beyond a long distance without interfering with the signals sent by the good nodes, the attackers have a dedicated, speed-of-light communication channel. The attackers have a total control over the wormholes. They can choose to tunnel a packet through or drop it without knowing the content of the packet.

3.4.5 Notations

For the clearance of the remainder of the chapter, we give out the following notations: If pairwise keys are applied, K_{AB} and K_{BA} are same and both represent the secret key between node A and node B. $MAC_K(M)$ represents the MAC code computed over message M with key K. If digital signatures are applied, $Sign_A(M)$ represents the signature of A over message M. All nodes that have the public key can verify the signature.

Every node can locate its geographic position. The position of node A is P_A. The maximum error of the distance between two positions is δ. If node A and node B read their positions at the same time, the real distance between them $d_{AB} \leq ||P_A - P_B|| + \delta$. The clocks are loosely synchronized. The difference between the clocks of any two nodes is smaller than Δ. We also assume that v is the upper bound of the velocity of node movement.

3.5 Detecting wormhole attacks

3.5.1 Basic end-to-end mechanism

Design goals

In the end-to-end wormhole detection mechanism, we only assume that the source and the destination of a route trust each other. This assumption holds under most conditions. If on-demand routing protocols are used, the source and the destination can negotiate the parameters such as the frequency to send wormhole detection packets through the routing request and reply. If geographical routing protocols are used, the destination can predetermine the parameters and submits them to the location servers. The source will get them when acquiring the position of the destination.

As shown in Section 3.2, wormholes need to be examined repeatedly during the route's lifetime. The detection information can be attached to the routing packets or the data packets. If the frequency of data packets is too low, wormhole detection packets can be sent separately. For multiple applications that use the same route, only one group of detection packets need to be sent. No detection is required for a route if it is no longer in use.

Compared to the mechanisms that only detect closed wormholes, the end-to-end mechanism has two special features:

First, every intermediate node will attach its timestamps and positions to the detection packets, but all examining operations are conducted by the destination node. In this way it can detect the conflicting information sent by the attacker to different neighbors. However, a scheme must be designed to prevent the destination node from being overwhelmed by the overhead.

Second, cross packet examination must be adopted by the end-to-end mechanism. Examining the packets independently can miss some wormholes. For example, in the half open wormhole shown in Figure 3.1b, $M1$ can declare that it receives a packet at the position of $M1$ and forwards it at the position of $M2$. As long as the distance d_{M1M2} and the declared buffering time t satisfy $d_{M1M2} \leq v \times t$, examining the single packet cannot find any anomaly.

In the cross packet examination, if a node declares its position P_1 at its clock time t_1, and P_2 at its clock time t_2, the destination can estimate its average moving speed and examines whether it is lying. If

$$\frac{||P_1 - P_2|| - \delta}{||t_1 - t_2|| + \Delta} > v \tag{3.1}$$

the node lies about its positions and there is a wormhole on the path.

Delivery of detection packets

We assume that the pairwise keys have been deployed. Every node A on the path has a secret key K_{AD} with the destination D. D knows the path length in hops and the identity of every node along the path. This can be achieved through the routing protocols such as DSR [7] or the explicit attachment of node identify to the routing packets.

When a wormhole detection packet is sent from the source, it contains seven fields: source address S, destination address D, the message M, a sequence number id for this route, the local time t_{Ssend} when the packet is sent, the position P_{Ssend} at that time, and the MAC code $MAC_{K_{SD}}(\ S, D, M, id, t_{Ssend}, P_{Ssend})$. M can be a routing packet, a data packet, or void if the detection packet is sent separately. Every time after sending a detec-

$$S: \quad h_s = MAC_{K_{SD}}(S, D, M, id, (t_{Ssend}, P_{Ssend}))$$
$$S \rightarrow A: \ (S, D, M, id, (t_{Ssend}, P_{Ssend}), h_s)$$

$$A: \quad h_A = MAC_{K_{AD}}(Received\ Packet, (t_{Arecv}, P_{Arecv}), (t_{Asend}, P_{Asend}))$$
$$A \rightarrow B: \ (Received\ \ Packet, (t_{Arecv}, P_{Arecv}), (t_{Asend}, P_{Asend}), h_A)$$

$$B: \quad h_B = MAC_{K_{BD}}(Received\ Packet, (t_{Brecv}, P_{Brecv}), (t_{Bsend}, P_{Bsend}))$$
$$B \rightarrow D: \ (Received\ \ Packet, (t_{Brecv}, P_{Brecv}), (t_{Bsend}, P_{Bsend}), h_B)$$

Figure 3.2. Delivery of a wormhole detection packet in end-to-end mechanism

tion packet for this route, the source increases id by 1. Examining the received ids, D can find out how many detection packets have been dropped.

When an intermediate node A forwards the packet, it attaches two $\langle$time, position$\rangle$ pairs: $\langle t_{Arecv}, P_{Arecv}\rangle$ when it receives the packet, and $\langle t_{Asend}, P_{Asend}\rangle$ when it forwards the packet. Then it attaches $MAC_{K_{AD}}(M_A)$, where M_A includes both the received packet and the two attached pairs. Figure 3.2 shows an example of the delivery procedure. A three-hop path is assumed.

Two potential problems exist in the proposed mechanism. First, how can a node get the accurate sending time and calculate it into the MAC code? Second, how does the signal propagation time affect the detection? For the first problem, the end-to-end mechanism has a weaker requirement on the accuracy of the sending timestamp than packet leashes. So we can adopt either of the two approaches [22] for temporal leashes.

For the second problem, taking the signal propagation time t_{prop} into account, we have the following equation for the sending time t_{send} at this node and the receiving time t_{recv} at the next hop:

$$||t_{recv} - t_{send}|| \leq \Delta + t_{prop} \tag{3.2}$$

If the direct communication range between neighbors is known, we can estimate $\Delta' = \Delta + t_{prop}$. The sending time and the receiving time of the same transmission will always satisfy:

$$||t_{recv} - t_{send}|| \leq \Delta' \tag{3.3}$$

Detection operations at the destination

When D receives a detection packet, it will check the following items: (a) All MAC codes are calculated correctly. (b) The neighboring nodes are within the direct communication range when the packet is passed. (c) The average moving speed of a node between it receives and forwards the packet does not exceed v. (d) The sending and receiving time of the same transmission satisfy equation 3.3. (e) The new $\langle$time, position$\rangle$ pair and the previous pairs of the same node do not conflict as shown in equation 3.1.

Items (a), (b), (c) and (d) focus on a single packet. The MAC codes guarantee the authenticity and integrity of the $\langle$time, position$\rangle$ pairs. Since the MAC codes cover the whole packet (including the information attached by previous nodes), it is very difficult for the attacker to record a part of the packet and conduct resend attacks. If the attacker resends the whole packet, it is the same as the duplicate packet generated by the network. Item (b) will detect the closed wormholes on the route.

Item (e) implements the cross packet examination. Through calculating the average moving speed of the intermediate nodes using their current and previous $\langle$time, position$\rangle$ records, the destination node monitors their movements. Items (c), (d) and (e) together prevent the malicious node from cheating the wormhole detection mechanism by declaring fake positions.

The wormhole detection packets may get lost because of the unreliable network. They can also be intentionally dropped or corrupted by the malicious nodes. Using the sequence number, the destination node can monitor how many detection packets have been lost. If a wormhole is detected, or q consecutive detection packets are all lost, the destination node will broadcast a message which notifies the source to abort the current route and activate the reinitiation. The relation between the value of q and the position information density is discussed in Section 3.6. The same condition will happen if a route becomes expired and no more detection packets are sent. Under this condition, the source will ignore the broadcast message.

Compared to previous approaches, the end-to-end mechanism does not let the intermediate nodes verify the neighbor relations by themselves. On the contrary, all examinations are conducted by the destination. The proposed mechanism can detect closed wormholes because the good nodes at different ends of the tunnels will not lie about their positions. For half open and open wormholes, if the malicious nodes send their real positions in the detection packets, the fake neighbor connections will be discovered because they are longer than the direct communication range. To avoid being detected, a smart attacker can buffer the packets and declare that it moves to the other end of the wormhole and forwards the data. This attack can be prevented by introducing packet lifetime into the network. It will restrict the longest moving distance of a node during the delivery of a single packet so that the length of the tunnel will be limited. It will also guarantee the position information density of the intermediate nodes as shown in Section 3.6.

Restricted by the positioning and clock synchronization errors, the end-to-end mechanism could introduce false alarms into the system. For example, the attacker will be able to tunnel the packets δ beyond d without being detected if the destination considers the positioning error when calculating the distance between two points. On the contrary, if the destination ignores the error, some connections having a distance close to d will be wrongly accused as wormholes. With the progresses in the positioning and synchronization techniques, the error rate will become smaller.

3.5.2 Overhead of basic end-to-end mechanism

Since the mobile devices have limited resources, the end-to-end mechanism, as a security enhancement, must consider the communication, computation, and storage overhead.

Every intermediate node attaches two $\langle$time, position$\rangle$ pairs and a MAC code to each detection packet. If pairwise keys are used, it has been shown in [22] that a PDA can accomplish 220K times MAC code calculation in one second. So there is not much computation overhead at the intermediate nodes. And they do not need to store any information.

The communication overhead includes byte overhead and packet overhead. If we use an eight-byte timestamp (if the time unit is $1ns$, it covers more than 500 years), an eight-byte position (on the surface of the Earth, it locates a position within $0.1m$), and an eight-byte MAC code, every intermediate node will attach 40 bytes to the packet. If the maximum transmission unit (MTU) is 1,500 bytes and M is 1,000 bytes, twelve nodes can attach their information.

In this way the byte overhead will increase fast to the path length. It does not scale well to long routes. To control the byte overhead, the destination node can choose a part of the intermediate nodes to attach their information. It changes the selected nodes to guarantee that everyone is examined. If k consecutive nodes on the route are chosen, it is the same as an end-to-end detection on these k nodes. More details of this method and its impact on the detection capability are studied in Section 3.6. The packet overhead is determined by the frequency to send the detection packets. This question is also studied in Section 3.6.

All examining operations are conducted by the destination node. We assume that the path length is l, and m packets carrying the detection information arrive at the destination. It needs $O(l)$ operations to examine a single packet. For every intermediate node, there will be $2m$ $\langle$time, position$\rangle$ pairs. If the cross packet examination calculates the average moving speed between every two pairs, there will be $O(m^2)$ operations for each node. So the total computation overhead for the m packets will be $O(lm + lm^2)$.

For the same reason, the destination needs to store $2m$ $\langle$time, position$\rangle$ pairs for every intermediate node. The required storage space will be $O(lm)$.

The lm^2 entry in the computation overhead and lm entry in the storage space put challenges to the mobile nodes with limited resources. For example, if a route is ten-hop long and 1,800 wormhole detection packets are received, the destination needs to store 36K $\langle$time, position$\rangle$ pairs (about 580 Kbytes), and conducts 65M operations. As the number of routes increases, the node cannot afford the required resources. In the next section, we propose a mechanism called COTA that requires $O(c_1 l)$ space and $O(c_2 lm)$ computation overhead (where c_1 and c_2 are constant), but having the same wormhole detection capability as the end-to-end mechanism.

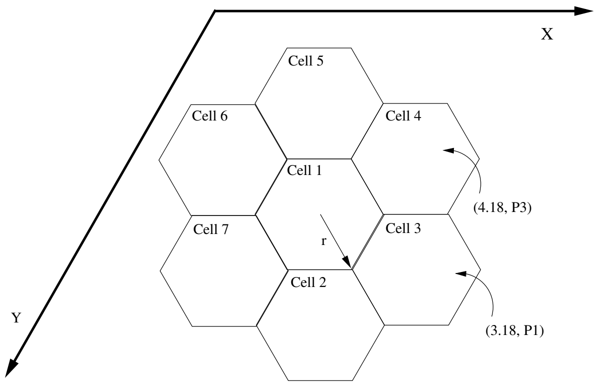
3.5.3 Cell-based open tunnel avoidance

COTA avoids to record and compare all $\langle$time, position$\rangle$ pairs. It divides the whole area into same-sized cells (hexagon), and divides the time into same-length slots. COTA only stores the first received $\langle$time, position$\rangle$ pair of every node that falls into the same cell and the same slot. Through adjusting the cell size and slot length, a node can control the efforts that it wants to put on the wormhole detection. The whole structure is a three dimensional cube of $\langle$time, position$\rangle$ records, and the format of the index is $\langle$node identity, cell number, slot number$\rangle$. When a new $\langle$time, position$\rangle$ pair of a node arrives, the destination selects from each cell a record of that node that has the shortest time difference from the new timestamp and calculates the average moving speed. In this way some pairs belonging to the same node are not compared. In Section 3.5.4 we prove that after adding an offset to equation 3.1, COTA has the same detection capability as the end-to-end mechanism.

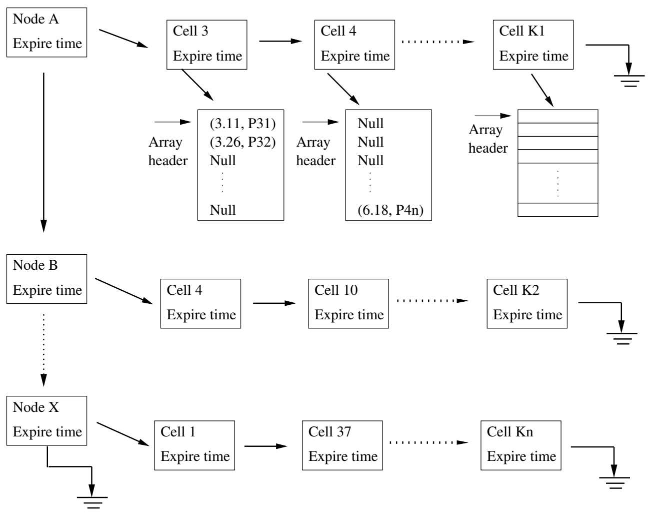
We also define the lifetime T_{life} of a packet. If a packet has traveled in the network for a time longer than T_{life}, it will be discarded by the destination node. Since the clocks are loosely synchronized, COTA can estimate the packet traveling time. To guarantee that the packet arrives at the destination D within its lifetime, the sending time at S and the receiving time at D should satisfy $(T_{Drecv} - T_{Ssend}) \leq T_{life} + \Delta$.

Two advantages have been brought to COTA by the definition of packet lifetime. First, it restricts the number of time slots that COTA needs to store for every intermediate node. If the slot length is T, the destination node only needs to store at most $(T_{life} + \Delta)/T + 1$ records for every intermediate node in every cell. Second, it restricts the longest moving distance of a node during the delivery of a single packet. It prevents the attacker in an open or half open wormhole from buffering the packet for a long time and declaring that it moves to the new position and forwards the packet.

An example of cell division and the data structure is shown in Figure 3.3. We assume that the time slot is 100 ms. For one intermediate node, the destination will only store one record in every slot in a cell. Now assuming that another record of node A with the content $\langle t_1$ = 3.18 sec, position = $P_1\rangle$ is received, and it falls into cell 3. Since there is already a



(a) Cell division of network area



(b) Data structure organization

Figure 3.3. Cell division and data structure of COTA in a destination node

record of node A in cell 3 in the slot 3.1 – 3.2 second, the new entry will not be recorded. Then the destination will select from each cell the record of node A that has the shortest time difference from the new timestamp. In this example, they will be (3.11, P_{31}) in cell 3 and (6.18, P_{4n}) in cell 4. If the selected pair is represented as $\langle t_2, P_2 \rangle$, D will calculate the average moving speed of node A between these two positions by

$$\bar{v} = \frac{max(0, ||P_1 - P_2|| - \delta)}{||t_1 - t_2|| + \Delta} \tag{3.4}$$

If $\bar{v} > v$, we know that A sent false information and there is a wormhole on the route.

Now let us consider another example. A record $\langle t_3$ = 4.18, position = $P_3\rangle$ of node A is received, and it falls into cell 4. There is no record of node A in cell 4 in the slot 4.1 – 4.2 second. So the new entry will be recorded by the destination. Then D will select the record of node A from each cell that has the shortest time difference from the new timestamp and calculates the average speed. In this case, they will be (3.26, P_{32}) in cell 3 and (6.18, P_{4n}) in cell 4.

COTA only stores and compares a part of the $\langle$time, position$\rangle$ records to reduce the storage and computation overhead. There are two questions that we have to ask: (1) can COTA detect all anomalies that the end-to-end mechanism can detect? (2) how much space and computation do we save? We discuss the answers in the next two sections.

3.5.4 Detection capability of COTA

COTA simplifies the end-to-end mechanism to reduce the storage and computation overhead. But it may miss the detection of some anomaly. An example is given out as follows. We have four $\langle$time, position$\rangle$ pairs of node M_1 as $\langle t_{M_11}, P_{M_11}\rangle$, $\langle t_{M_12}, P_{M_12}\rangle$, $\langle t_{M_13}, P_{M_13}\rangle$, and $\langle t_{M_14}, P_{M_14}\rangle$. The first two pairs fall into cell 1 and slot 1. The second two pairs fall into cell 2 and slot 2. We calculate the average moving speed between every two pairs using equation 3.4. The speed between pair two and four is faster than v, but the speed between any other two pairs is not. If we record and compare all pairs, we will find the anomaly and detect the wormhole. However, if COTA is applied and pair one and three arrive first, they will be recorded. Later when pair two and four arrive, they will not be stored and they are never compared to each other. COTA does not detect this anomaly.

However, adding an offset to equation 3.4 will enable COTA to detect all wormholes that the end-to-end mechanism can detect. The proof is given as follows.

Lemma 1 *If COTA uses*

$$\frac{max(0, ||P_{new} - P_{select}|| - \delta + 2r + vT)}{||t_{new} - t_{select}|| + \Delta} \tag{3.5}$$

to calculate the average moving speed between the new and the selected $\langle$time, position$\rangle$ pairs, it can detect all wormholes that can be detected by the end-to-end mechanism. In *equation 3.5, δ is the maximum error of the distance, r is the radius of cells, v is the highest speed of nodes, T is the length of a time slot, and Δ is the clock error.*

Proof We assume that we have two pairs, $\langle t_1, P_1\rangle$ and $\langle t_2, P_2\rangle$, of the same node that show the anomaly

$$\frac{||P_1 - P_2|| - \delta}{||t_1 - t_2|| + \Delta} > v \tag{3.6}$$

And without losing generality, we assume that $\langle t_1, P_1\rangle$ is received by the destination first. When $\langle t_2, P_2\rangle$ arrives, there are two possible conditions:

Condition I: $\langle t_1, P_1\rangle$ is recorded by COTA.

If $\langle t_1, P_1\rangle$ is selected to be compared with $\langle t_2, P_2\rangle$, the average moving speed (after adding $2r + vT$) is faster than v, and the anomaly will be detected.

If it is not selected, there must be another pair $\langle t_3, P_3\rangle$ that falls into the same cell as P_1 but has a shorter time difference from t_2. So we have $||t_3 - t_2|| \leq ||t_1 - t_2||$. Since P_3 and P_1 are in the same cell, the longest distance between the two positions is $2r$. So we have:

$$\begin{aligned} ||t_3 - t_2|| &\leq ||t_1 - t_2|| \\ ||P_3 - P_2|| + 2r &\geq ||P_1 - P_2|| \end{aligned} \tag{3.7}$$

Combining equation 3.6 and 3.7, we must have

$$\frac{||P_3 - P_2|| - \delta + 2r}{||t_3 - t_2|| + \Delta} > v \tag{3.8}$$

Condition II: If $\langle t_1, P_1\rangle$ is not recorded by COTA, there must be another pair $\langle t_3, P_3\rangle$ that is recorded. It falls into the same cell and same slot as $\langle t_1, P_1\rangle$, but it arrives earlier. So we have:

$$\begin{aligned} ||P_3 - P_2|| + 2r &\geq ||P_1 - P_2|| \\ ||t_3 - t_2|| - T &\leq ||t_1 - t_2|| \end{aligned} \tag{3.9}$$

If $\langle t_3, P_3\rangle$ is selected to be compared with $\langle t_2, P_2\rangle$, combining equation 3.6 and 3.9, we will get:

$$\frac{||P_3 - P_2|| - \delta + 2r + vT}{||t_3 - t_2|| + \Delta} > v \tag{3.10}$$

If $\langle t_3, P_3 \rangle$ is not selected to be compared with $\langle t_2, P_2 \rangle$, there must be another recorded pair $\langle t_4, P_4 \rangle$ that falls into the same cell but has a shorter time difference. So we have:

$$||t_4 - t_2|| - T \leq ||t_3 - t_2|| - T \leq ||t_1 - t_2|| \tag{3.11}$$

Combining equation 3.6 and 3.11, we will get

$$\frac{||P_4 - P_2|| - \delta + 2r + vT}{||t_4 - t_2|| + \Delta} > v \tag{3.12}$$

Combining all conditions shown in equation 3.6, 3.8, 3.10, and 3.12, we have:

$$\frac{||P_{select} - P_{new}|| - \delta + 2r + vT}{||t_{select} - t_{new}|| + \Delta} > v \tag{3.13}$$

■

After introducing the offset $2r + vT$ into COTA, we guarantee that it can detect all wormholes that the end-to-end mechanism can detect. We define $2r + vT$ as the $sensitivity$ of COTA. An optimistic node can use COTA as in equation 3.4. It will allow all neighbors that are within the direct communication range to exchange packets. But in some cases, a real attacker is able to tunnel a packet as far as $2r + vT$ beyond the range. A more conservative node can use the format shown in equation 3.5. It will guarantee the detection capability, but in some cases it will introduce false positive alarms. The impact of false alarms will be studied through simulation in Section 3.8.

3.5.5 Parameter determination

There are three parameters in COTA: packet lifetime, cell size, and time slot length. The choices of the parameters impact not only the detection capability of COTA, but also the storage and computation overhead. We now discuss them separately.

The choice of packet lifetime is directly related to the end-to-end delay in ad hoc networks. Estimating this delay accurately is a difficult problem because it is affected by traffic load, path length, movement model, network size, and other features. There have been theoretical analyses of this problem in simplified network scenarios [71–74].

The results provide valuable guidelines for the design of protocols. However, the various simplifying assumptions do not always hold in real network settings.

Another approach is to conduct real measurements. The results collected from simulations have been shown in [75–78]. The typical value of the average delay is a few hundred milliseconds in the studied network scenarios. In this chapter, we assume the similar network conditions, thus the similar average delay is also assumed. The analysis in [77] shows that after the mode of the curve, the probability density function of delay decreases exponentially as time increases. To control the fraction of detection packets that are discarded because of unexpected long delay, we set T_{life} an order of magnitude longer than the average delay. The typical value of T_{life} is a few seconds. When an approach for more accurate delay estimation appears, we can replace this method without impacting other components of COTA.

Now let us consider the required storage space and computation operations of COTA. We assume that the packet lifetime is T_{life}. The destination node only needs to record the $\langle$time, position$\rangle$ pairs that were sent in the past $T_{life} + \Delta$. During this period, the possible position of a good node must be within a circle with the diameter $v(T_{life} + \Delta)$ (use the position when $t = (T_{life} + \Delta)/2$ as the center, and $v(T_{life} + \Delta)/2$ as the radius). If there are no wormholes, the number of cells that have active records for the node is at most

$$\frac{\pi(v(T_{life} + \Delta)/2)^2}{1.5\sqrt{3}r^2}. \tag{3.14}$$

In each cell, at most $(T_{life} + \Delta)/T + 1$ records are stored. So the total number of records stored by the destination for one node is at most

$$\begin{aligned} &\frac{\pi v^2(T_{life} + \Delta)^2}{6\sqrt{3}r^2}(\frac{T_{life} + \Delta}{T} + 1) \\ &\simeq \frac{\pi v^2(T_{life} + \Delta)^3}{6\sqrt{3}r^2 T} \end{aligned} \tag{3.15}$$

If the path length is l, the destination needs to store at most $l\times$equation 3.15 records for one route.

When a new record arrives, the destination will select from each cell a record of that node to compare with it. There are at most $(T_{life} + \Delta)/T + 1$ records in each cell for the node. If a linear search is applied, the new record needs at most

$$\frac{\pi(v(T_{life} + \Delta)/2)^2}{1.5\sqrt{3}r^2} \times (\frac{T_{life} + \Delta}{T} + 1) \tag{3.16}$$

operations to accomplish the cross packet examination. If a balanced tree is used to store the records in each cell for a node, the examination in one cell can be accomplished in $\log_2((T_{life} + \Delta)/T + 1) + 1$ operations. But the maintenance overhead for the tree structure may outrun the benefits. In the following analysis, we assume that the linear search is applied. If there are m COTA packets for the route and the path length is l, the total number of operations required by COTA is

$$2ml \times equation\ 3.16 \tag{3.17}$$

We examine a practical example to show the savings of COTA. We assume that r is 12 m, T_{life} is 5 seconds, v is 20 m/s, T is 200 ms, l is 10 hops, and 1,800 wormhole detection packets are received. The destination node needs to store at most 9 K $\langle$time, position$\rangle$ pairs. If a linear search is used to locate the record in every cell, the destination needs to conduct 32.5 M operations. Compared to the overhead of the end-to-end mechanism shown in Section 3.5.2, COTA saves 75% on space and 50% on computation. Examining equation 3.15 and 3.16, we find that when $m > \frac{\pi v^2(T_{life}+\Delta)^3}{6\sqrt{3}r^2T}$, COTA will save in both space and computation. As the number of wormhole detection packets increases, COTA can save more because the required space does not change and the computation overhead increases at a linear speed.

If the $sensitivity$ of COTA, $2r + vT$, is pre-determined, we can choose suitable values of r and T to minimize the required storage space and the computation overhead. From equation 3.15 and 3.16, we find that both requirements are minimized at the same values of r and T. Figure 3.4 shows the consumed space for one intermediate node for different $sensitivity$ values. It can also be viewed as the curve of computation overhead.

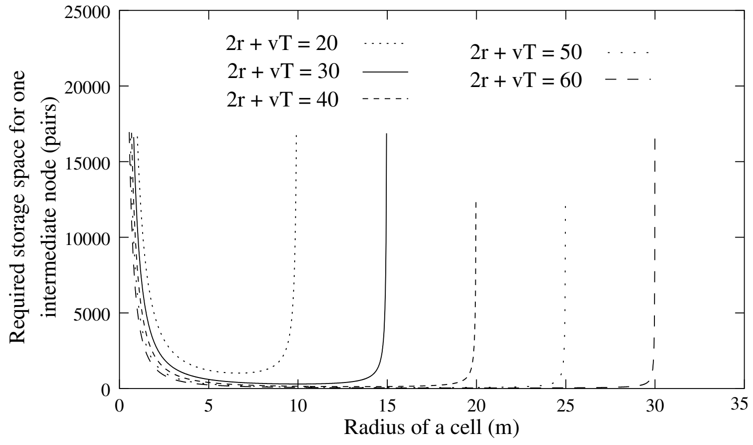


Figure 3.4. Storage space of COTA with different values of $sensitivity$

We assume that $2r + vT = X$, where X is a constant. Equation 3.15 is minimized when

$$\frac{\partial}{\partial r}(r^2(X - 2r))^{-1} = 0 \tag{3.18}$$

The optimal value of r equals to $\frac{1}{3}X$. It shows that the best choices of the cell size and the slot length increase linearly when the value of $sensitivity$ becomes larger. This allows a mobile node to pre-determine the resources that it wants to consume to defend against wormhole attacks. If we take the results back, we find that COTA has achieved its design goals: for every intermediate node, a constant storage space and the computation overhead that increases linearly to the number of the detection packets.

From Figure 3.4 we find that the storage and computation overhead of COTA is relatively stable within a large interval of r. Therefore, when the chosen r and T are biased from the optimal values, the overhead does not increase sharply. Since the resources available to mobile nodes vary greatly, this enables the nodes to make flexible choices on the parameters. Table 3.1 compares the overhead between the optimized COTA and the end-to-end mechanism. It assumes a 10-hop route and 1,800 wormhole detection packets. The values of T_{life} and v are the same as before.

Table 3.1
Overhead comparison between the end-to-end mechanism and COTA

	Basic end-to-end		Optimized COTA	
sensitivity	storage(pair)	operation	storage(pair)	operation
30m	36 K	65 M	5.20 K	18.66 M
40m	36 K	65 M	2.25 K	8.12 M
50m	36 K	65 M	1.12 K	4.04 M
60m	36 K	65 M	0.70 K	2.49 M
70m	36 K	65 M	0.43 K	1.57 M

3.5.6 Data structure maintenance

COTA uses a complex data structure. If it cannot be efficiently maintained, the overhead caused by the insertion/deletion of records and the collection of unused space will cancel out all benefits described in previous sections. We are going to answer two questions about COTA: (1) How to organize the stored records? (2) How to efficiently recollect the space of the expired records?

A practical data structure in the destination node is shown in Figure 3.3b. All nodes that have active records are put into a link list. The destination remembers the expiration time of the latest $\langle$time, position$\rangle$ pair of every node. It is used in space collection. Every node has a link list, which stores the cells that have active records. Each cell also remembers the expiration time of the latest pair in it. In each cell, an array of records are maintained. The size of the array is $(T_{life} + \Delta)/T + 1$. Every entry has a bit to identify whether the data is available in it. To avoid memory copy as time elapses, the array is used in a cyclical way and a pointer is used to identify the current array header.

When a new $\langle$time, position$\rangle$ pair of a node arrives, the destination first locates the cell list of that node. Then for each cell of that node, it uses a linear search to locate the record that has the shortest time difference from the new timestamp.

This linear search is also used to abort those expired records and to recollect the space if all records in that cell have expired. It resets the available bits of those expired entries and moves the array header according to current time. If the latest pair in a cell has expired, COTA can take the memory back without looking into the array because all records must have expired.

This space collection method works well if new records of a node keep arriving. On the contrary, if a node is not on any active routes and no new records for it arrive, the space occupied by it cannot be efficiently recollected. To overcome this difficulty, the destination node needs to periodically traverse the node list to examine the expiration time of the latest record of every node. If the latest record has expired, all space occupied by that node can be recollected.

The maintenance of the data structure will increase the space requirement within a limited range. If all pointers are four-byte and timestamps are eight-byte, the required space will increase less than 10%. Resetting the available bits and moving the array headers can be accomplished when COTA traverses the array of each cell. They do not add much computation overhead. Referring to the results shown in Table 3.1, we find that COTA can still save a lot of resources.

3.5.7 Experiments on real devices

In this section, we present some results on the computation efficiency of COTA. They are collected through experiments on real mobile devices. We assume that symmetric cryptographic operations are used.

We define the procedure that locates the record in an array who has the shortest time difference from the new timestamp and calculates the average moving speed as a $unitcheck$. It corresponds to the operations that examine a new record in one cell. It also clears out the expired records in that cell. We design a test program that executes 100 million $unitchecks$ and run it on a Compaq iPAQ 3630 with 206 MHz CPU and 64M RAM. Different values of $sensitivity$ are examined and the size of an array is determined by the optimal value of the time slot length. The computation efficiency of COTA is shown in Table 3.2.

Table 3.2
Computation efficiency of COTA

Sensitivity (m)	20	30	40	50	60
Optimal size of an array	19	13	10	8	7
$Unitcheck$ / sec	1.05M	1.54M	2.01M	2.54M	2.86M

For example, when the $sensitivity$ is 30 m, the device can execute 1.54 million $unitchecks$ in one second with the optimal values of cell size and time slot length. Therefore, as the destination of a 10-hop route that receives 5 wormhole detection packets per second, the node will use 0.28% of its CPU to check the records in these packets. If it uses 5% of its CPU to conduct wormhole detection, it can support 17 such routes simultaneously.

3.6 Controlling communication overhead

The design of COTA allows a mobile node to pre-determine the storage and computation resources that it wants to put on wormhole detection. In this section, we focus on the communication overhead. We study packet overhead and byte overhead separately.

3.6.1 Frequency to conduct wormhole detection

Wormhole detection needs to be conducted repeatedly during the lifetime of a route. The detection frequency determines the packet overhead. It also impacts the intervals between the received $\langle$time, position$\rangle$ pairs of the intermediate nodes.

An intermediate node cannot control the longest buffering time of a detection packet in it. However, the source and the destination can determine the frequency to send such packets when the route is established. Through adjusting this frequency, the destination can control the longest interval between the received timestamps of an intermediate node. If the interval between the timestamps of the source in any two consecutive detection packets is shorter than t_{int}, and at least one of q consecutive detection packets will reach to the destination, the longest interval T_{int} between the received timestamps of the source satisfies $T_{int} \leq qt_{int}$. If the lifetime of a packet is T_{life}, we have:

Lemma 2 *If the longest interval between the received timestamps of the source is T_{int}, the longest interval between the received timestamps of any intermediate node is $T_{int} + T_{life} + 2\Delta$. Here Δ is the error of clock, and T_{life} is the packet lifetime.*

Proof Let's consider the $\langle time, position \rangle$ pairs of an intermediate node A received by the destination. The pairs received in the same detection packet have the same sub-index. If we randomly select a $\langle time, position \rangle$ pair, there are two possible conditions:

Condition I: the selected pair records a receiving event. We assume that it is $\langle t_{Arecv1}, P_{Arecv1} \rangle$. There must be a record, $\langle t_{Asend1}, P_{Asend1} \rangle$, that remembers the sending of this packet. And we have $t_{Asend1} \epsilon (t_{Arecv1}, t_{Arev1} + T_{life}]$. Therefore, when searching in the direction that the time increases, we must be able to find another record of A within the interval T_{life}.

The longest interval between the received timestamps of the source is T_{int}. Therefore, there must be a received record of the source $\langle t_{SourceSend}, P_{SourceSend} \rangle$ that satisfies $t_{SourceSend} \, \epsilon \, [t_{Arecv1} - T_{life} - T_{int} - \Delta, t_{Arecv1} - T_{life} - \Delta)$. The receiving and sending timestamps of this packet at node A must be smaller than t_{Arecv1}. Considering the clock error, when searching in the direction that the time decreases, we must be able to find one record of A within the interval $T_{life} + T_{int} + 2\Delta$.

Condition II: the selected pair records a sending event. Similar analysis can be conducted.

Therefore, the longest interval between the received timestamps of any intermediate node is $T_{int} + T_{life} + 2\Delta$. ■

Since the position information is received together with the timestamps, this interval also determines the density of the position information of an intermediate node. For example, if the interval is 10 seconds and T_{life} and Δ are set as before, T_{int} = 3 seconds. If the probability that a packet gets lost or corrupted on the path is P, the probability that all q detection packets are lost is P^q if the events are independent. For example, if the probability that a detection packet gets lost is 25% and q = 5, the probability that five consecutive wormhole detection packets are all lost is less than 0.1%. The source needs to send one detection packet every 0.6 second. The end-to-end mechanism will not add packet overhead if the data packet density is larger than 2 packet/second.

3.6.2 Controlling byte overhead

The analysis in Section 3.5.2 shows that the end-to-end mechanism does not scale well to very long routes if every intermediate node attaches its $\langle$time, position$\rangle$ pairs. To avoid this problem, the destination can choose a part of the nodes to attach their records. The selected intermediate nodes are switched to guarantee that every neighbor relation is examined. For example, the destination can ask the first $\lceil \frac{l}{2} + 1 \rceil$ nodes on the route to attach their records for q detection packets, then the second $\lceil \frac{l}{2} + 1 \rceil$ nodes on the route will attach their records for q detection packets. The two halves of the nodes are overlapped to avoid the detection gap. In this way the byte overhead will decrease for more than 50%. However, the longest interval between the received timestamps of an intermediate node will become $2T_{int} + T_{life} + 2\Delta$. This may allow a wormhole to exist for a longer time before it is detected.

3.7 Security analysis

Because of the error of position, the error of clock, and the $sensitivity$ of COTA, there exist false alarms in the end-to-end mechanism. If the node adopts a conservative policy, some real neighbor relations will be considered as wormholes. The destination will abort the route and activates the routing reinitiation. This leads to the increase in routing overhead and the average delay. On the contrary, if an optimistic policy is adopted, the real neighbor relations will not be wrongly accused. However, the attackers will be able to tunnel the packets beyond the direct communication range without being detected. In Section 3.8 we study both conditions through simulations.

A malicious node can eavesdrop the wormhole detection packets. The position information attached by the intermediate nodes is protected by the MAC codes. The malicious node cannot send fake information in other node's name to fabricate a wormhole on the route. Every intermediate node calculates the MAC code based on the whole packet. An attacker cannot take a part of the detection packet and conducts the resend attack. If it resends the whole packet, it is the same as a duplicate packet generated by the unreliable network. For multiple connections that have the same source and destination and use the same route, only one group of wormhole detection packets need to be sent. A malicious node cannot overwhelm the destination by establishing multiple connections at the same time. The adoption of COTA allows the destination to control the overhead on each route. It helps to defend against the Distributed Deny of Service (DDoS) attacks generated by a group of collusive attackers.

A wormhole tries to fabricate a non-existent route. The malicious node cannot drop the detection packets to avoid being discovered. If a violation is detected in the received information, the destination will broadcast the anomaly condition. The source receiving this message will try to establish a new route. When a wormhole is detected, the end-to-end mechanism does not try to identify the attacker. Therefore, the malicious node cannot frame a good node.

3.8 Simulation and results

We study the practical impacts of the proposed wormhole detection mechanism through simulation. The experiments are deployed using ns2 [6]. Two properties are of special interest: false alarm ratio and communication overhead.

3.8.1 Simulation setup

We assume that pairwise keys have been deployed. AODV [5] is chosen as the routing protocol and is updated to combine with the detection mechanism. When the source initiates the routing request (RREQ) packet, it proposes its choices of the parameters such as the $sensitivity$ and the frequency to send out wormhole detection packets. When an intermediate node forwards the routing request, it will attach its identity. The source and the intermediate nodes will protect the attached information using keyed hash codes. When the destination receives the request, it knows the identities of all intermediate nodes. The destination reviews the proposed parameters. If they satisfy its requirements, the destination will accept them. Otherwise, it will propose its choices in the routing reply (RREP). Besides the parameters, the destination will also determine the starting value of the sequence number for the detection packets. When the source receives the reply, it will start to send out the data packets and the detection packets.

RREQ and RREP accomplish the route discovery and try to settle the wormhole detection parameters. There are chances that the source and the destination cannot agree with each other on the parameters. A separate round of negotiation can be conducted after the route is established and before the data packets are transferred. A method similar to the negotiation procedure in the Internet Key Exchange (IKE) protocol [79] can be adopted.

Table 3.3 lists the simulation parameters of ns2. The node moving speed covers a range from human jogging to vehicle riding in the country field. We assume that the moving speeds of the nodes are uniformly distributed from 0 to the highest value. The sources and the destinations of the connections are randomly picked from the mobile nodes. For each examined condition, five connection scenarios and four node movement scenarios are generated and the average values of the simulation results are shown in figures.

The parameters of the detection mechanism are as follows: T_{life} = 5 seconds, T_{int} = 3 seconds, Δ = 1 second, and δ = 10m. The longest interval between two consecutive detection packets sent by the source is 0.6 second, which is a little longer than the average interval between data packets. Therefore, most of the wormhole detection information

Table 3.3
Simulation parameters

Simulation duration	1000 seconds
Simulation area	1000 * 1000 m
Number of mobile nodes	50
Transmission range	250 m
Movement model	Random waypoint
Highest node speed	8 – 20 m / s
Number of connections	30
Traffic type	CBR (UDP)
Data payload	512 bytes
Packet rate	2 pkt / s
Node pause time	0 second

can be attached to data packets. We define the longest distance that a node can move in one second as the unit distance. It can be calculated as $v \times 1second$. Different values of $sensitivity$ (from 0.5 to 2.5 times of the unit distance) are examined.

Since the wormhole detection mechanism is a security enhancement, its false alarm ratio is of special interest. Both false positive and false negative mistakes are studied. The storage and computation overhead has been discussed extensively in Section 3.5. In this part we focus on the communication overhead in packets and bytes.

3.8.2 Results on false alarms

Figure 3.5 and 3.6 illustrate the false positive alarms in an environment where no wormhole exists. In Figure 3.5, all destination nodes adopt the proposed mechanism as in equation 3.5 to guarantee the detection capability. The curves show the relation between the number of false alarms and the $sensitivity$. Four values of the highest speed, from

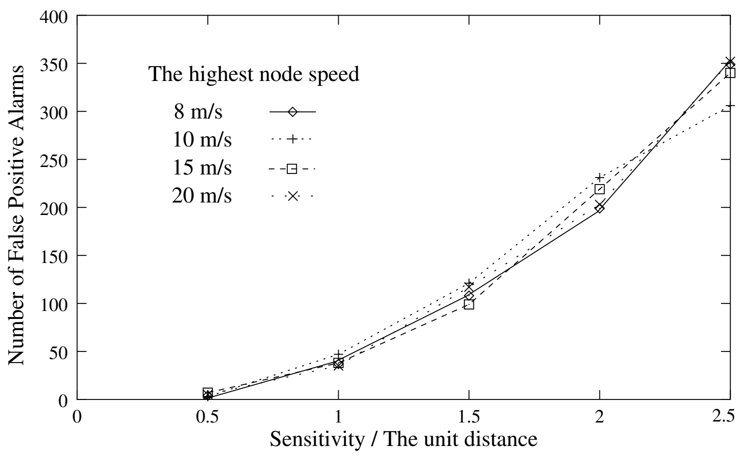


Figure 3.5. False positive alarms: no wormholes, sensitivity / unit distance changes

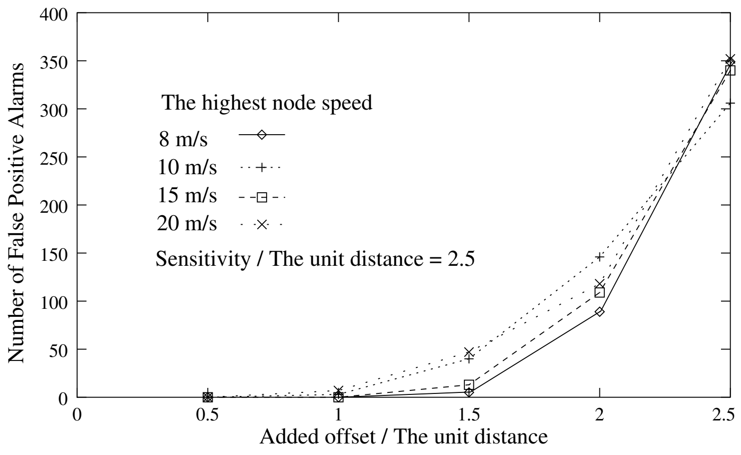


Figure 3.6. False positive alarms: no wormholes, added offset / unit distance changes

8m/s to 20m/s, are examined. The values of r and T are determined to minimize the computation and storage overhead. The value of the $sensitivity$ increases from 0.5 to 2.5 times of the unit distance. From Figure 3.5, we find that as the ratio increases, the number of false alarms increases faster than a linear function. Another interesting point is that the curves for different speeds stay close to each other. In the proposed mechanism, it is the ratio between the $sensitivity$ and the unit distance, instead of the absolute value of it, that determines the number of false positive alarms.

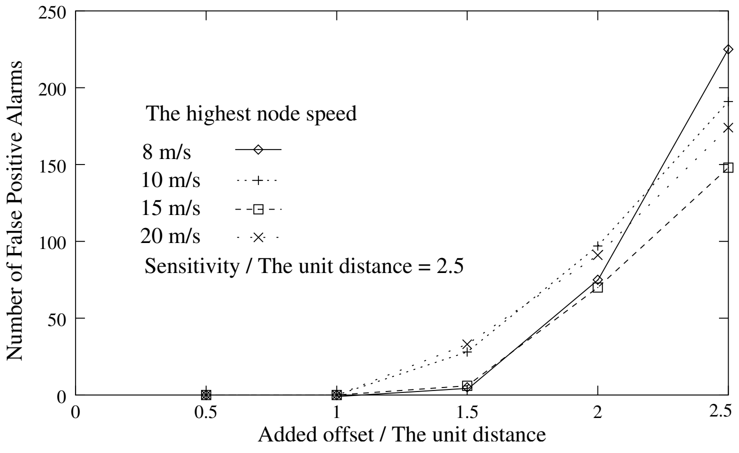


Figure 3.7. False positive alarms when wormholes exist in the network

False positive alarms lead to breaks of existent routes. To reduce the number of such mistakes, equation 3.5 can be updated to the following format:

$$\frac{max(0, ||P_{new} - P_{select}|| - \delta + offset)}{||t_{new} - t_{select}|| + \Delta}$$

in which the added offset is a value between 0 and the $sensitivity$ to achieve different tradeoffs between the detection capability and false alarm ratio. Figure 3.6 illustrates the relation between the number of false positive alarms and the added offset. The network setup is the same as in Figure 3.5. The ratio between the $sensitivity$ and the unit distance is 2.5. The number of false alarms increases as the added offset increases. Compared to Figure 3.5, fewer false positive alarms are introduced in Figure 3.6. The curves for different speeds are close to each other.

When the added offset is smaller than the $sensitivity$, it causes fewer false positive mistakes. However, as the analysis shows in Section 3.5, it may also miss the detection of some real wormholes. The following experiments are conducted to study this impact. Among the 50 nodes in the network, two pairs of nodes are randomly selected as "potential attackers". Each pair has a long-range, out-of-band wireless channel that can be used to construct a wormhole. The attackers' channels do not interfere with each other or the channel used by the good nodes. To construct the wormholes that cannot be detected when the added offset is 0, the longest distance that the attackers can tunnel beyond the direct

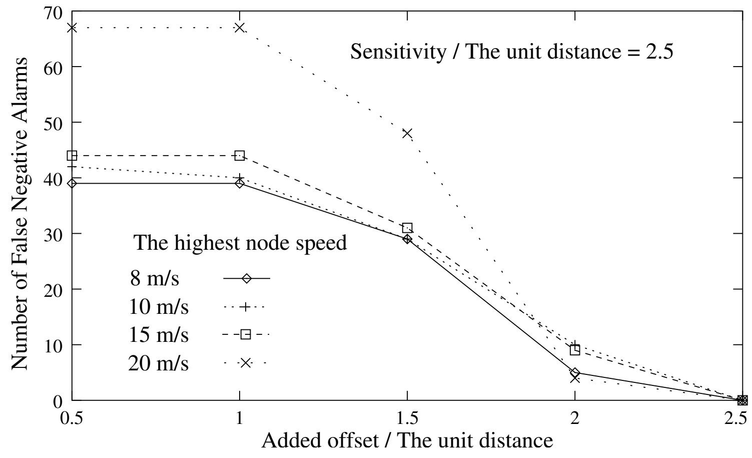


Figure 3.8. False negative alarms when wormholes exist in the network

communication range is the *sensitivity*. Therefore, the attackers will form a wormhole only when the distance between them falls into that interval. Other simulation parameters are the same as in Figure 3.6. Figure 3.7 and 3.8 illustrate the false positive and negative mistakes as the added offset changes.

In Figure 3.7, the curves are similar to those in Figure 3.6. Since some real wormholes are introduced into the system, fewer false positive mistakes are made. In Figure 3.8, as the added offset increases, fewer and fewer real wormholes are missed by the detection mechanism. Through adjusting the choices of the *sensitivity* and the added offset, a mobile node can achieve a better tradeoff between the false alarm ratio and the detection capability.

3.8.3 Results on communication overhead

The following experiments explore the extra communication overhead introduced by the wormhole detection mechanism. We study both packet overhead and byte overhead. To establish the baseline for the comparison, we also examine the overhead caused only by the routing protocol when the detection mechanism is not enabled. Figure 3.9 and 3.10 illustrate the results collected from an environment in which no wormholes exist.

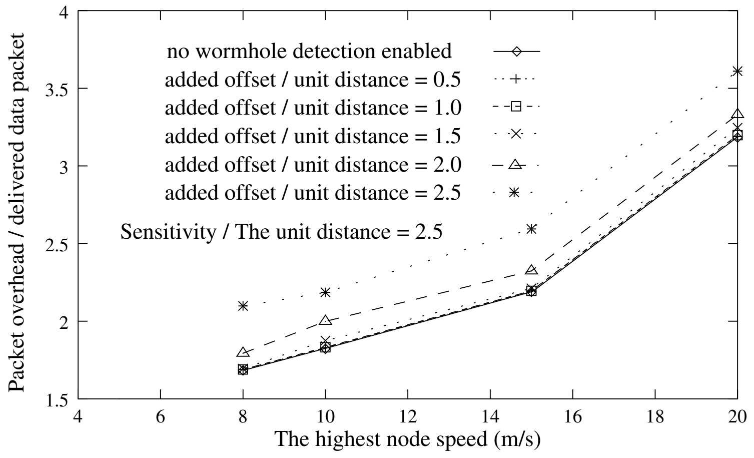


Figure 3.9. Packet overhead of the wormhole detection mechanism

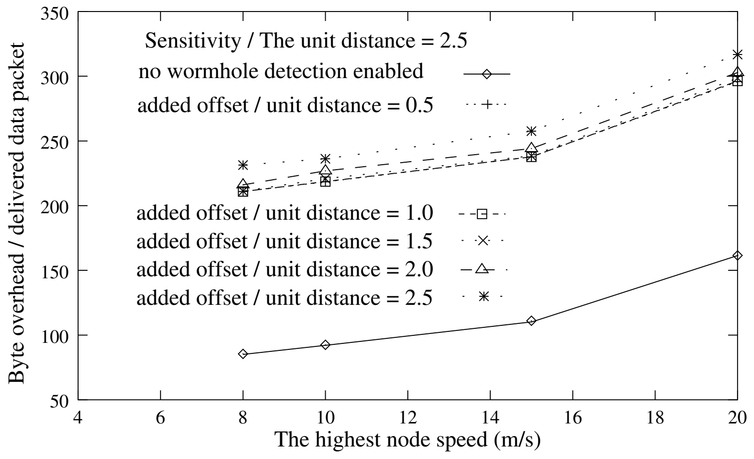


Figure 3.10. Byte overhead of the wormhole detection mechanism

As stated in Section 3.8.1, most of the detection information is attached to the data packets. The extra packet overhead is primarily caused by the route rediscovery procedure after the detection of a “wormhole”(could be a false alarm). From Figure 3.9 we find that as the added offset increases, more packet overhead is introduced. But the increase is small when the ratio between the added offset and the unit distance is small. Through adjusting the values of $sensitivity$ and the added offset, a node can control the increase of packet overhead.

The increase in byte overhead shown in Figure 3.10 is more notable. The extra bytes are primarily caused by the $\langle$time, position$\rangle$ pairs attached by the intermediate nodes, so the number of false alarms does not impact them to a large extent. The curves for different values of the ratio are close to each other. But compared to the overhead when no wormhole detection is enabled, the increase is sharp. The justifications for this increase are two folds. First, the overhead is still reasonable for the applications in which the wormholes must be prevented. The communication peers can lower the relative byte overhead by increasing the length of data packets. Second, since the routes in the simulation are not long, we do not enable the byte overhead control method stated in Section 3.6.2. The adoption of that method may help the nodes to improve the efficiency.

3.9 Discussions

The analysis shows that the $sensitivity$ represents a tradeoff between the detection capability and the required resources. Many features can impact the choice of its "optimal" value and here we only explore a few of them. First, the $sensitivity$ is restricted by the positioning accuracy because two records having a distance shorter than δ could represent the same point in the network. Second, the $sensitivity$ is impacted by the movement patterns of the nodes. The example in Section 3.5.4 shows that COTA misses some anomalies because the records falling into the same slot and the same cell might be ignored. Therefore, taking the movement patterns into account, we can choose a suitable $sensitivity$ value and determine r and T to control the occurrence of such events. More experiments will be conducted to explore their relations so that we can predict the optimal interval of the $sensitivity$ given a certain scenario.

As illustrated in Section 3.5.3 and 3.6.1, the definition of packet lifetime T_{life} can restrict the number of stored records at the destination node. The other contribution of T_{life} is to guarantee the record density of the intermediate nodes. It can prevent the attacker in an open or half open wormhole from declaring that it moves back and forth between two ends of the wormhole and forwards the packets. If the proposed mechanism can be integrated with IDS, this attack can be detected by examining the anomaly in the node movement patterns. Under that condition, the restriction of T_{life} can be loosened and replaced by a more generic time window duration.

In the proposed mechanism, wormhole detection is conducted in a pro-active manner. We may expect that a reactive wormhole detection mechanism will cause less communication and computation overhead. To enable this update, the mechanism needs to be combined with the intrusion detection systems so that it will be activated when suspicious conditions are discovered.

In the proposed mechanism every mobile node acts individually to detect wormholes. The computation overhead and detection accuracy can be further improved if the nodes can share the knowledge securely and cooperate on the detection operations. Mechanisms will be designed to enable this information exchange procedure and to verify the authenticity of the information. The mechanisms can also be applied to defend against wormhole attacks conducted by a group of collusive attackers.

Geographic based wormhole detection can be viewed as an example of the emerging Location Based Services (LBS). One security concern of LBS is the disclosure of the location and movement patterns of a mobile node. For example, in COTA, the destination has the real time location information of the source and every intermediate node. This may conflict with the privacy concerns of some users. A potential extension is to use a "shadow" position to replace the real item while keeping the distance among nodes the same. The work is motivated by the research in multi-dimensional scaling [80]. It allows the nodes to take advantages of LBS while achieving privacy preservation.

3.10 Conclusion

The classification of wormhole attacks on ad hoc networks constructs a basis on which the detection mechanisms can be examined and compared. It divides the attacks into three groups: closed, half open, and open. The previous solutions focus on the prevention of closed wormholes. This leads to the requirement of a more generic approach.

An end-to-end mechanism is presented that can detect closed, half open and open wormholes. To reduce the storage and computation overhead, we present a new scheme, COTA, to manage the detection information. It records and compares a part of the $\langle$time, position$\rangle$ pairs. With a suitable relaxation, COTA has the same detection capability as the end-to-end mechanism. Through adjusting the cell size and time slot length, a node can control the resources that it wants to put on wormhole detection.

The schemes to control communication overhead are studied. Through adjusting the frequency to send detection packets, the longest interval between position information of any intermediate node is guaranteed. To improve the scalability of the proposed mechanism, the destination can select a part of the nodes to attach their wormhole detection information.

The practicability of the proposed mechanism is examined using both simulation and experiments on real mobile devices. The false alarm ratio can be controlled by adjusting the parameters such as the $sensitivity$ and the added offset. When the $sensitivity$ is 30m, a Compaq iPAQ 3630 with 206M Hz CPU and 64M RAM can use 0.28% of its CPU to accomplish the wormhole detection on a 10-hop route. COTA does not depend on a specific authentication mechanism. It can be combined with other approaches such as TESLA to construct new wormhole detection mechanisms.

The immediate extensions to our work consist of two parts. First, we propose to combine the mechanism with the location based routing protocols. Second, new schemes to reduce the detection packet frequency and byte overhead are under development. They will lead to a generic, efficient approach that helps the ad hoc networks to defend against the wormhole attacks.

4 VISUALIZATION OF WORMHOLES IN SENSOR NETWORKS

4.1 Introduction

The malicious nodes can conduct wormhole attacks on sensor networks. Since the sensors use a radio channel to send information, the malicious nodes can eavesdrop the packets, tunnel them to another location in the network, and retransmit them. This generates a false scenario that the original sender is in the neighborhood of the remote location and it will mislead the understanding of the local network topology. An example of the wormhole attack on an underwater sensor network is illustrated in Figure 4.1.

Compared to the nodes in ad hoc networks, the sensors usually have less resource on computation capability, network bandwidth, and storage. Therefore, the wormhole attacks will impact the sensor network performance to a larger extent. The simulation results in [24, 81] show that when there are more than two wormholes in the network, more than 50% of the data packets will be attracted to the fake connections and get discarded.

The previous approaches for wormhole detection in ad hoc networks [22–24] usually require the mobile nodes to be equipped with some special hardware, such as positioning devices, synchronized clocks, or directional antennas. Although the progresses in integrated circuit design and hardware manufacture will make these devices become cheap, small, and power efficient enough to fit in sensors in the future, an approach that does not require any special hardware is expected at this stage.

In this chapter, we present a family of approaches, visualization of wormholes, to defend against such attacks on sensor networks. First, we study the problem in a static, two dimensional sensor network. The proposed mechanism, MDS-VOW (Multi-Dimensional Scaling - Visualization Of Wormhole), does not require the sensors to be equipped with special hardware. It reconstructs the network using multi-dimensional scaling and detects the wormhole by visualizing the anomalies introduced by the attack.

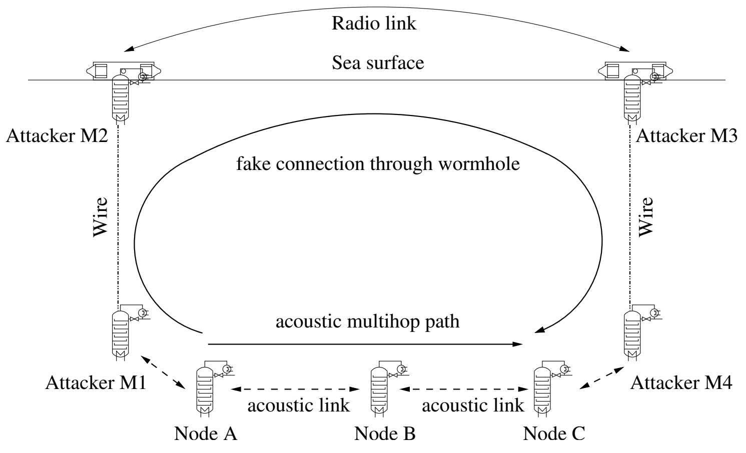
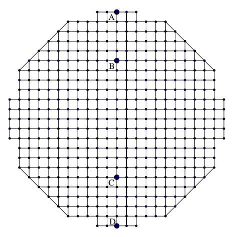


Figure 4.1. Wormhole attacks in underwater sensor networks

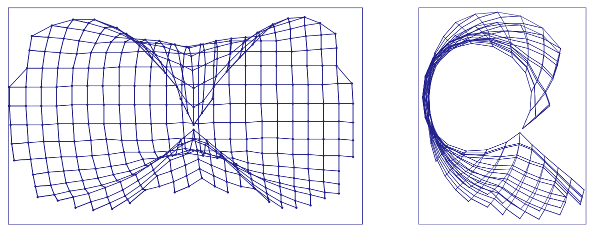
MDS-VOW consists of four steps: (1) It uses the inaccurate distance measurements between the sensors that can "hear" each other (might through a wormhole) as inputs to estimate the distance between every sensor pair. (2) Using multi-dimensional scaling, we reconstruct the network of sensors and calculate a virtual position for each of them. (3) A surface smoothing mechanism is adopted to compensate the impacts of distance estimation errors on the reconstructed network. The mechanism will preserve the features that are introduced by the wormhole. (4) The shape of the reconstructed network is analyzed and the fake neighbor connections will be identified.

For wormhole detection in a more complicated scenario, we study the problem in underwater sensor networks. Two features are of special interest: (1) the slow propagation speed of acoustic signals in water, (2) the node movement and topology changes caused by water current. We propose a distributed approach, Dis-VoW (Distributed Visualization of Wormhole), to defend against such attacks. Every sensor will collect the distance estimations from its neighbors and reconstruct the local network within two hops using multi-dimensional scaling. Dis-VoW then uses the distortions in edge lengths and angles among neighbor sensors in the reconstructed networks to detect the wormholes.

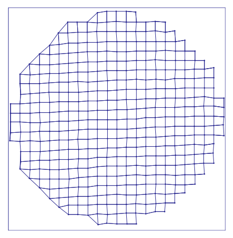
Dis-VoW consists of four steps: (1) Every sensor will estimate the distances to its neighbors using the round-trip time of the acoustic signals. (2) Through broadcasting these



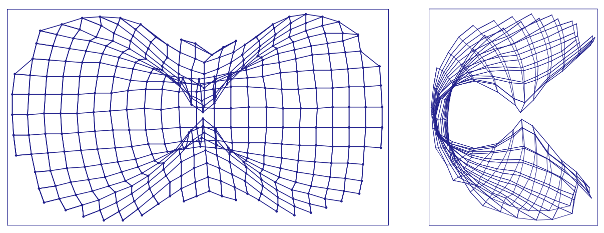
(a)



(c) a wormhole links sensor A and C



(b)



(d) a wormhole links sensor B and C

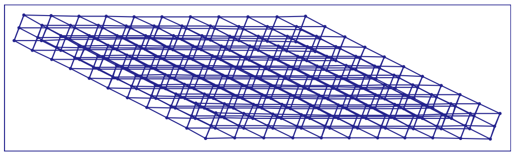
Figure 4.2. Impacts of wormholes on global reconstruction of 2D networks

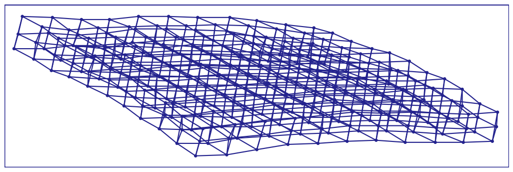
distances, every sensor will be able to use MDS to reconstruct the local topology within two hops. (3) Every sensor will examine the reconstructed local network. If distortions are discovered, wormhole detection method will be activated so that the fake neighbor connections can be located. (4) The detected wormholes will be avoided during routing discovery and packet forwarding so that the network safety and performance is preserved.

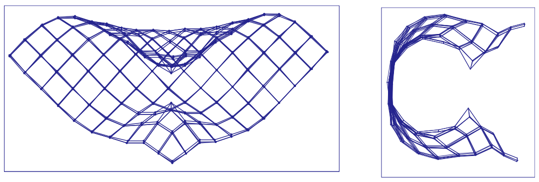
Before presenting the details of the mechanisms, we use several examples to illustrate the impacts of wormholes on network reconstruction using MDS in Figure 4.2 and 4.3. Figure 4.2a shows the original, static, 2D sensor network and the sensors are deployed on the grids in a circle area. Figure 4.2b shows the reconstructed network using MDS when no wormhole exists and its layout is almost the same as the original network. In Figure 4.2c and 4.2d, the wormhole will pull the sensors at the two ends to each other through the fake connection, and results in a bent surface. Through detecting the bending feature, MDS-VOW will identify the fake connections and locate the wormhole.

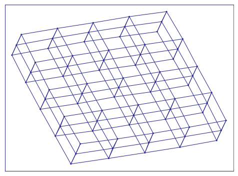
The impacts of wormholes on the reconstruction of a 3D network are illustrated in Figure 4.3. Figure 4.3a shows the original layout when the sensors are placed as a $11 \times 11 \times$

3 grid. A part of the neighbor connections are shown as lines to assist the understanding of the topology. Figure 4.3b shows the global reconstruction of the network using MDS when there is no wormhole in the network. The layout is almost the same as the original network. In Figure 4.3c, a wormhole links two sensors that are faraway from each other and MDS bends the reconstructed network to fit the fake connection. In Figure 4.3d and 4.3e, we show the reconstruction result of a local network within two hops to a sensor, without and with a wormhole in it, respectively. In the reconstructed layouts, we can easily see the distortions caused by the fake neighbor connection. Dis-VoW will allow every sensor to detect the distortions and use them to locate the wormhole.


(a) original sensor layout: a $11 \times 11 \times 3$ grid.


(b) global reconstruction with no wormholes.


(c) global reconstruction with one wormhole.


(d)

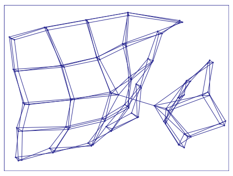

(e)

Figure 4.3. Impacts of wormholes on global and local reconstruction of 3D networks

As we will demonstrate, the proposed mechanisms can effectively identify the fake neighbor connections. The contributions can be summarized as follows: (1) They do not require the sensors to be equipped with any special hardware. Therefore, the deployment of the methods will not increase the hardware costs. (2) Techniques from social science, computer graphics, and scientific visualization are integrated to solve network security problems. (3) In Dis-VoW, every sensor reconstructs the network and detects the wormhole locally. Therefore, the computation and storage overhead is affordable for a weak node such as a sensor. The reconstructed network is an arbitrary rotation and translation of the original layout. Since the detection methods focus on the shape of the network instead of the coordinates of the sensors, we do not require the deployment of “anchors” that are equipped with positioning devices.

The remainder of the chapter is organized as follows: In Section 4.2, we review the related work. Section 4.3 and 4.4 describe the details of MDS-VoW and Dis-VoW, respectively. Section 4.5 and 4.6 present the experimental results acquired through simulation. Two scenarios, grid placement and random placement of the sensors, are studied. Section 4.7 discusses the impacts of node density, safety of the proposed mechanisms, and the frequency to conduct wormhole detection. Section 4.8 concludes the chapter.

4.2 Related work

4.2.1 MDS and its applications in wireless networks

Multi-dimensional scaling was originally a technique developed in the behavioral and social science for studying the structure of objects. The inputs to MDS are the measures of the difference or similarity between object pairs [82]. The output of MDS is a layout of the objects in a low-dimensional space. In this chapter, the input is the distance matrix between the sensors. The mechanism can reconstruct the network and calculate a virtual position for each sensor. We adopt the classical metric MDS in the proposed mechanisms, in which the distances are treated as in a Euclidean space. More details of MDS can be found in [80, 82].

MDS has been applied to solve the localization and positioning problems in wireless networks. In [83], a solution using classical metric MDS is proposed to achieve localization from mere connectivity information. The algorithm is more robust to measurement errors and requires fewer anchors than previous approaches. A distributed mechanism for sensor positioning using MDS has been presented in [84]. It develops a multi-variate optimization-based iterative algorithm to calculate the positions of the sensors. Another approach [85] for sensor network localization applies semi-definite programming relaxation to minimize the errors for fitting the distance measurements.

4.2.2 Wormhole detection in ad hoc networks

Different approaches that adopt positioning devices [22], accurate clocks [23], or directional antennas [24] have been proposed. A detailed description of these methods can be found in Chapter 3.

4.2.3 Underwater sensor networks

There has been an increasing interest in monitoring the marine environment for scientific exploration, commercial exploitation, and coastline protection. The ideal vehicle for these applications is a scalable underwater sensor network, which employs a large number of distributed, unmanned, and untethered wireless nodes to locally gather information in a timely manner. Some preliminary research has been conducted in [86–88]. This self-organizing, self-reconfigurable network provides effective supports in sensing, monitoring, and reconnaissance.

The underwater sensor network paradigm is different from the existing land-based sensor networks. The acoustic link features a long latency and a low bandwidth, while the sensor nodes are with low or medium group mobility due to water current. Extended research is required for security enforcement in this scalable environment to protect the localized sensing and coordinated networking.

4.2.4 Distance estimation between wireless nodes

Several mechanisms have been developed to estimate the distance between wireless nodes. The proposed solutions include received signal strength [89], Time-of-Arrival and Time Difference of Arrival [23, 90, 91], and triangulation [92, 93].

Among them, Received Signal Strength Indicator is the most cost-efficient method because it does not require any extra hardware on the node. One disadvantage, however, is that the measured distance can be inaccurate. For example, an estimation error from 5% to 40% of the radio range has been assumed in [94]. The estimation accuracy can be improved by establishing a more accurate signal propagation model or using the stability of the strength difference at various points. In MDS-VOW, we adopt a mechanism from computer graphics to smooth the reconstructed network and compensate the impacts of the measurement errors.

Since the acoustic signals transfer very slowly in water (about $1500m/s$), the sensors can use a not-so-accurate clock to measure the round-trip time of a packet transfer and estimate the distance between two nodes. We assume that the clock drift does not exceed a maximum value ρ. If in real world the length of a time duration is t , and C is the value measured by a sensor, we have: $1 - \rho \leq \frac{dC}{dt} \leq 1 + \rho$. The clocks embedded in wireless nodes can easily achieve a relative accuracy of $\rho = 10^{-6}$ [95]. If the direct communication range of the acoustic channel is $150m$, the measurement error of the round-trip time is shorter than $0.2\mu s$. We have developed a single round protocol to estimate the upper bound and lower bound of the distance between two sensors. The protocol is discussed in Section 4.4. More details and the correctness and robustness analysis can be found in [81].

4.2.5 Key distribution in sensor networks

During the distance estimation procedures, both the challenges and the replies should be protected by the secret keys. To avoid one compromised sensor leading to the collapse of the whole system, we do not assume the adoption of group communication keys. Therefore, either pairwise keys or pre-distributed secrets can be adopted to protect the information. If pair-wise keys are used, the secrets can be determined before the sensors are deployed [69]. The disadvantages are the fast increase of the number of keys and the difficulty to update them when new sensors are added. A random key pre-distribution approach for sensor networks is proposed in [96], which allows any pair of sensors to share a key with a certain probability. Various approaches have been proposed to improve the safety and key sharing probability. The adopted methods include $q-$composite and multi-path key reinforcement [97], Co-operative protocol [98], the usage of Blom's key pre-distribution scheme [99], pseudo random function, and bivariate polynomials [100].

4.3 MDS-VOW: A centralized approach

4.3.1 System assumptions

We assume that the links among sensors are bidirectional and two neighbor sensors can always send packets to each other. This assumption will hold under most conditions when the power of the sensor has not been exhausted. We assume that two sensors are neighbors when the distance between them is shorter than r, where r is defined as the radio range.

To use MDS to reconstruct the network layout, we assume that the sensors and their neighbor relations construct a connected graph, for example, a path exists between every sensor pair. The density of sensors may also impact the network reconstruction, and we give more discussion on this problem in Section 4.7.

We assume that a special node exists in the sensor network, which is called the "controller". The controller can accomplish the $O(n^3)$ operations required by the MDS mechanism in a short period of time when there are n sensors in the network. In our experiments, we use a PC with 1.8G CPU and 512M RAM as the controller. When $n \approx 400$, it takes the machine a few seconds to reconstruct the network. When the controller broadcasts a message with full power, all sensors in the network can receive the data. On the contrary, only the sensors within the radio range to the controller can directly communicate to it. Others need to send their packets through the multi-hop routes. The adoption of the centralized controller impacts the scalability of MDS-VOW. Extending the mechanism to a distributed approach is discussed in the next section.

We assume that the sensors are not self-movable. Therefore, once deployed, the route changes are mainly caused by the "dead" or broken of the sensors. Extending MDS-VOW to a movable environment is discussed in the next section.

The sensors broadcast the neighbor discovery beacons at the same power level so that the neighbors can estimate the distances using the received signal strength. The sensors report the list of nodes that they can hear and the estimated distances to the controller. We assume that the controller and the sensors share a group key. This key is only used to protect the traffic generated by MDS-VOW and it is mainly used during network bootstrap. Therefore, the time duration and the amount of traffic that a malicious node can acquire to break the key is limited. The key can be determined before the sensors are deployed [69].

The sensors estimate the distances with the received signal strength. The accuracy of the estimation is impacted by various factors, such as the terrain, background noise level, or even the weather condition. We model the measurement errors as uniform noises. With an error rate e_m, if the real distance between two sensors is d ($d \leq r$), a random value drawn from the uniform distribution $[d \times (1 - e_m), d \times (1 + e_m)]$ is used as the measured distance. If the chosen value is smaller than 0 or larger than the radio range, 0 or r will be used respectively. In our experiments, we examine different values of e_m from 0 to 0.8.

4.3.2 Network bootstrap

As we discussed before, the sensors send the list of nodes that they can hear and the distance estimations to the controller. The list may include both the real neighbors and the fake ones through the wormholes. Since the data is transferred through multi-hop routes and the detection has not been conducted, these routes could be controlled by wormholes and the data could become the target of the attacks. If the information cannot reach to the controller, the detection accuracy of MDS-VOW will be impacted. To prevent this problem, the network bootstrap is conducted as follows.

After sensor deployment, the controller will broadcast a route discovery packet to the sensors within the radio range r and mark the path length to itself as 0. The sensors receiving the packet will increase the path length by one and re-broadcast it. With every sensor remembers the previous hop, increases the path length by one, and re-broadcasts the packet, the routes to the controller will be established. If multiple packets are received by a sensor, it will use the one with the shortest path length. These routes can only be used to send the sensor lists and distance estimations for wormhole detection. Once the fake connections are excluded and the real neighbors are known to every sensor, other routing protocols can be adopted for sensing data transfer.

After measuring the distances to the sensors that it can hear and sending the information to the controller, every sensor will put itself in an "idle" state until a reply from the controller is received. In this state, a sensor will not forward packets for other nodes except the sensor lists and distance estimations. The controller examines the received packets and broadcasts a list with full power that includes the sensors whose information has not been received. These sensors, worrying about that their packets will get lost again, will flood the network with their information. The controller has a good chance to get the flooded packets. MDS-VOW will ignore the sensors whose packets are still not received.

For every sensor pair that can hear each other, the controller calculates the average of the two estimated distances and uses it in MDS. A connection will be ignored if none of the estimations or only one of them is received. Then the controller will execute MDS-VOW to identify the fake connections. It will send a reply with full power to every sensor and the reply includes the sensor's non-suspicious neighbors. After receiving the reply, the sensor will switch to the "operation" state and only uses the non-suspicious neighbors during route discovery and packet forwarding. Those neighbors that can be heard but fail to pass the detection will not be used. Therefore, a neighbor connection must be examined by MDS-VOW before it is adopted by the sensors. The attackers cannot hide the fake connections from the controller if they want the fake connections to be used.

After the execution of MDS-VOW, other routing protocols can be adopted to establish routes and transfer data as long as the sensors only use the non-suspicious neighbors. Since we assume that the sensors are not self-movable, there should not be any new neighbors appearing during the network operation. Therefore, MDS-VOW does not need to be run repeatedly when the sensors adapt to the route changes.

The packets transferred for MDS-VOW are protected by the group key. Message authentication code (MAC) can be calculated and attached to the packets to protect their integrity. Some sensors may stay in the "idle" state and cannot operate properly if they fail to get the replies. With the high density of sensors, their impacts on sensing coverage and routing are restricted.

4.3.3 Building blocks of MDS-VOW

Network reconstruction

The proposed mechanism uses the measured distances between the sensors that can hear each other to reconstruct the network layout. For every such pair, both sensors will estimate the distance and send it to the controller. The controller calculates the average value and puts the result at the suitable positions in the distance matrix. If the average value is larger than the radio range, r will be used in the matrix. The distance from a sensor to itself is 0. After the distances between the sensor pairs that can hear each other are calculated, a classical shortest-path algorithm, such as Dijkstra's algorithm, can be applied to calculate the shortest distance between every sensor pair. When all positions in the distance matrix have been filled, MDS can be applied to rebuild the network and a virtual position for every sensor will be generated.

Distance error compensation

The distance estimation errors have a significant impact on network reconstruction. As an example, Figure 4.4 shows the results of MDS when the sensors are deployed in a circle area as in Figure 4.2a. No wormhole exists in the network and the error rate e_m increases from 0.2 to 0.8. From the results, we find that mechanisms must be designed to compensate the errors while preserving the features that are introduced by the wormholes.

We propose to apply the smoothing algorithm for the reconstructed 3D surfaces [101, 102] to accomplish the task. The mechanism consists of two steps: (1) it calculates a fitting plane for every sensor based on the coordinates of itself and its neighbors, (2) a new position of the sensor is determined by the old coordinate and its projection on the fitting plane. The details are discussed as follows.

For a sensor whose position is s, the positions of its k neighbors are represented as N_0 to N_{k-1}. We first calculate the center of these $k+1$ nodes as:

$$c = \frac{s + \sum N_j}{k+1}, j = 0 \cdots k-1 \tag{4.1}$$

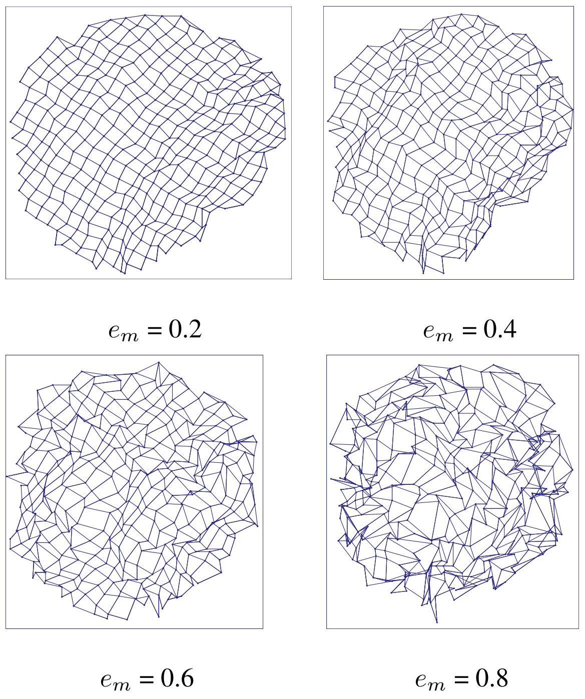


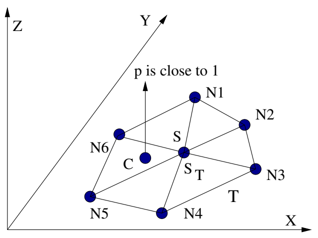
Figure 4.4. Impacts of distance measurement errors on network reconstruction

The fitting plane of s will pass the center. Besides a point on the plane, we also need to calculate its normalized vector v_s. Using the positions of these $k + 1$ sensors and the center, we can construct a 3 $\times$ 3 matrix as:

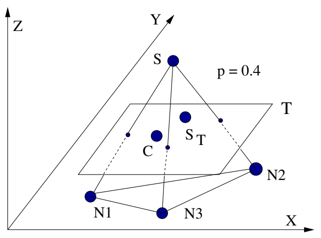
$$M = \sum_{j=0}^{k-1} (N_j - c)(N_j - c)^T + (s - c)(s - c)^T \tag{4.2}$$

M is a symmetric, positive semi-definite matrix and it has been shown in [102] that the unit eigenvector corresponding to the smallest eigenvalue of M is the normalized vector v_s of the desired plane. The eigenvalues can be calculated by the QR factorization [103]. With the calculation of c and v_s, the fitting plane T for s is determined.

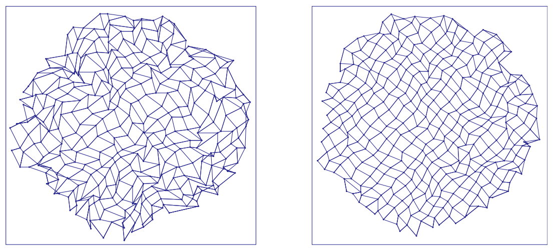
If two sensors s and s' are close to each other and the local surface is relatively flat, the fitting planes T and T' are nearly parallel. In other words, the point product of the two normal vectors has a value close to 1 or -1. On the contrary, if the surface close to s is very "bumpy", the normal vectors of T and T' may vary greatly. After calculating the fitting



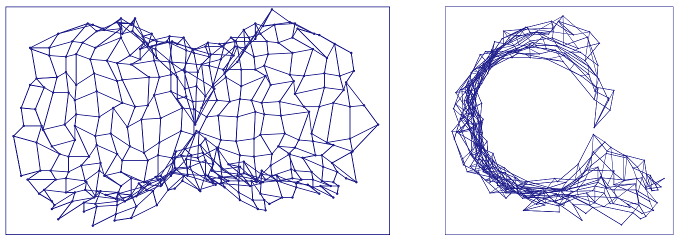
(a) The local surface is relatively flat.



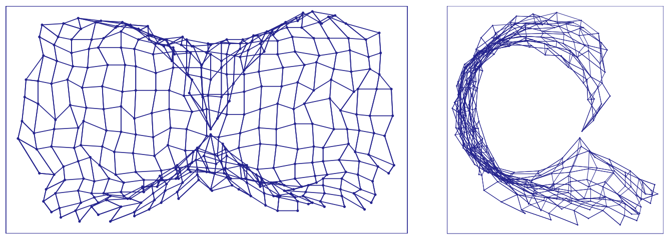
(b) The local surface is relatively bumpy.



(c) before smoothing after smoothing



(d) before smoothing



after smoothing

Figure 4.5. Smoothing the reconstructed network

planes, we generate a value p for every sensor to describe the smoothness of the nearby surface. For s and its neighbors N_0 to N_{k-1}, we define:

$$p_s = \frac{\sum |v_s \cdot v_{N_j}|}{k}, j = 0 \cdots k-1 \tag{4.3}$$

Since all vectors are normalized, p_s has a value between 0 and 1. The larger the value is, the more smooth the local surface is. The projection of s on T is represented as s_T. We smooth the reconstructed network by calculating the new position of s as:

$$new_s = p_s \times s + (1 - p_s) \times s_T \tag{4.4}$$

Examples of the smoothing procedure are shown in Figure 4.5. If the local surface is flat, c will also be on the same plane and p_s is close to 1. Therefore, new_s will almost be at the same position as s, as shown in Figure 4.5a. On the contrary, if the local surface fluctuates a lot, the normal vectors of the neighbors will point to all different directions. The new position will be close to s_T, as shown in Figure 4.5b. Different from the "bumpy" features caused by the measurement errors, the bending feature of the network is caused by the fake neighbor connections through the wormhole. Its impacted area is much larger than a sensor and its neighbors. It will not be removed by the smoothing operations that focus on a small area of the network and the results can be seen in Figure 4.5c and 4.5d.

Detection of wormhole

The results in Figure 4.5 show that if no wormhole exists in the network, the reconstructed surface after smoothing will be relatively flat. However, if two sensors are linked by a wormhole, the MDS mechanism will bend the reconstructed surface to fit the fake connection and minimize the fitting errors. If we imagine the network as a pie of soft rubber, the wormhole can be viewed as a "string" that pulls two sensors to each other and leads to the distortion of bending. Detecting the bending feature caused by the wormhole and locating the ends of the "string" will help us to identify the fake neighbor connections.

Figure 4.6 shows a reconstructed surface and the enlargement of the fake connection and its nearby areas. We find that the fake connection through the wormhole and the neighbor sensors at both ends will form a two-ended torch structure. This structure can be detected by examining the angles between the fake connection and the planes determined by the neighbor sensors. For example, the fake connection in Figure 4.6 is almost vertical to the plane $A_1A_3A_5$ and $B_1B_2B_3$. In MDS-VOW, we first derive a $normalized\ torch$

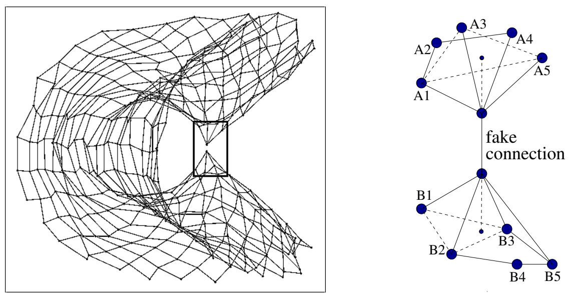
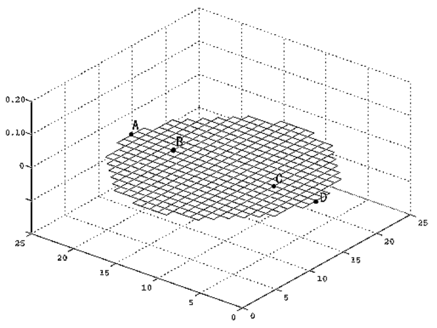


Figure 4.6. The two-ended torch structure caused by a wormhole

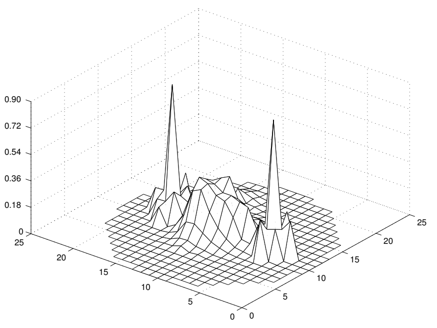
$counter$ for every connection based on the two-ended torch structures that it forms. Using the $counters$ of the connections that a sensor is involved in, we define a $wormhole$ $indicator$ for every sensor. MDS-VOW then labels the connections between two sensors with large $wormhole\ indicators$ as fake connections. The details are described as follows.

Assume that a sensor s can hear other k sensors as N_0 to N_{k-1}. For every connection sN_j $(j = 0 \cdots k-1)$, we calculate the number of two-ended torch structures that it forms. We assume that the neighbors of s can determine g different planes, and the neighbors of N_j can determine h different planes. We choose one plane from each set and examine the angles between the connection sN_j and the planes. When both angles are $\geq \frac{3\pi}{8}$, we count it as a two-ended torch structure. We examine all gh combinations. Since the number of the planes determined by the neighbors may vary greatly, the counter for sN_j is then normalized by dividing gh. We define this normalized number as the $normalized\ torch\ counter$ of the connection sN_j. When we have calculated the $counters$ for all the connections sN_j $(j = 0 \cdots k-1)$, we define the $wormhole\ indicator$ of s as $\max\{counter\ of\ sN_j, j = 0 \cdots k-1\}$. As the examples of the proposed mechanism, we demonstrate the $wormhole\ indicators$ of the sensors in different network scenarios in Figure 4.7.

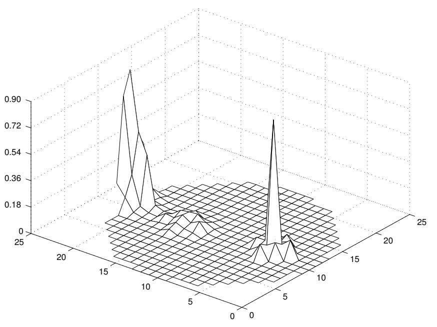
From Figure 4.7 we find that the sensors close to the ends of the wormhole can be easily identified. MDS-VOW then labels the connections between two sensors that have large *wormhole indicators* as fake connections. In our experiments, we set the threshold



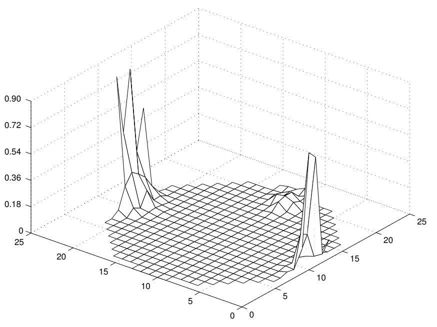
(a) No wormhole exists in the network



(b) A wormhole exists between sensor B and C



(c) A wormhole exists between sensor A and C



(d) A wormhole exists between sensor A and D

Figure 4.7. Wormhole indicators of the sensors in different network scenarios

at 0.6. The advantage of this method is that the sensors, no matter where their positions are in the network, can be handled in a uniform way. The disadvantage, however, is that some real neighbor connections may be wrongly accused as wormholes and false positive alarms will be introduced into the system. We study this problem in Section 4.5.

4.3.4 The MDS-VOW algorithm

With the readiness of all building blocks, we now walk through the steps of the MDS-VOW algorithm.

1. Every sensor estimates the distances to the nodes that it can hear and reports them to the controller.
2. The Dijkstra's algorithm is applied to calculate the distance between every pair of sensors and the distance matrix of the network is constructed.
3. Using the classical metric MDS method, MDS-VOW reconstructs the layout of the network and calculates a virtual position for each sensor.
4. Smoothing mechanism is applied to compensate the impacts of the measurement errors. The mechanism will preserve the feature that is introduced by wormhole.
5. The *wormhole indicator* of every sensor in the reconstructed network is calculated. The fake neighbor connections through wormholes are identified.
6. The fake connections are distributed to the sensors by the controller. These connections will be avoided during routing and packet forwarding.

4.4 Dis-VoW: A distributed approach

4.4.1 System assumptions

We study the distributed visualization of wormholes (Dis-VoW) in the underwater sensor networks. Two underwater sensors are considered as neighbors when the distance between them is shorter than r, where r is defined as the communication range. We assume that the links among sensors are bidirectional and two neighbor sensors can always send packets to each other. The assumption will hold when the power of the sensors has not been exhausted.

To use MDS to reconstruct the local network, we assume that the sensors are dense enough and there is no partition in the network, i.e. a path exists between every sensor pair. At the physical layer, we assume that omni-directional acoustic transmission and reception devices are adopted.

The computation complexity of MDS is $O(n^3)$, where n is the number of sensors in the reconstructed network. We have conducted some experiments and run the MDS program on a PC with 400MHz CPU and 256M RAM. We use two bytes to represent a distance between two sensors, which can provide an accuracy of $0.1m$ in a $1km^3$ cube area. When $n = 75$, the machine will use 11.25 KByte to store the 75×75 distance matrix and it takes the machine a few seconds to reconstruct the network. Moreover, if we adopt the key distribution methods discussed in Section 4.2, the integrity and safety of the distance measurement packets can be protected by symmetric encryption or one way functions. The computation overheads of these methods on real mobile devices have been studied in [22, 81]. Therefore, both the computation and storage overheads of the local reconstruction are affordable to a mobile device such as an iPAQ PDA.

Since most acoustic systems operate at the frequency below 30KHz, the available bandwidth of a channel is very limited. For example, the highest value reported so far is around 1Mbps at the range of $60m$ radius [104]. As surveyed in [105], research systems and commercial systems have highly variable link capacities but the attainable $range \times rate$ product can hardly exceed 40km-kbps. Long-range acoustic signals that operate over several tens of kilometers may have a capacity of only several tens of bits per second, while a short-range system operating over several tens of meters may have tens of kilobits per second. In our simulation, we set the communication range at $150m$ and the link rate is 200 kbps.

An important feature of underwater sensors is their moderate mobility. When the impacts of water current are considered, the sensors show a group movement as well as the changes of relative distances. The speed of ocean currents has been studied in various projects [106, 107]. The reported standardized speed ranges from 0.02 to $1.5m/s$. In our study, we set the sensor movement speed at $1-1.2m/s$ (about 2 knots). These settings can be replaced with minor efforts when a more accurate movement model of the underwater sensors becomes available.

4.4.2 Building blocks of Dis-VoW

Distance estimation between neighbor sensors

After deployment, every sensor needs to measure the distances to its neighbors so that the values can be used in local reconstruction. Since the propagation speed of acoustic signals in water is relatively slow, the clock drift has a very limited impact on the measurement accuracy. Therefore, we adopt a Time-of-Arrival approach to accomplish this task. We assume that two neighbor sensors have known each other's identity and a shared secret has been determined using the methods in Section 4.2. To reduce the impacts of node movements on the measurement accuracy, we propose a one-round protocol to determine the upper and lower bound of the distance between two sensors.

A $Prover\ P$ and a $Verifier\ V$ try to measure the distance between them and they share a secret key K_{PV}. We assume that the real distance between them is d_{PV}, the measured value is $\overline{d_{PV}}$, the signal propagation speed is v, the channel bandwidth is w, and symmetric encryption is adopted. The protocol executes as follows:

1. V chooses a random nonce x and uses the encryption function e to calculate the L'-bit challenge that includes $e_{K_{PV}}(x)$. V sends the challenge to P and starts its clock at local time t_0 when the first bit of the challenge is transferred.

2. P applies K_{PV} to decrypt the challenge and get x. Then P uses a one way hash function h to calculate the L-bit reply that includes $h(x)$ and sends it back to V.

3. V stops the local clock at t_1 when the last bit of the reply is received. It returns $T = t_1 - t_0$.

Using the hash result in the reply avoids the potential attacker to get both the plain text and cipher-text of x. If V sends out the challenge at t_0, the $Prover\ P$ will receive the challenge at $\frac{d_{PV}}{v} + \frac{L'}{w} + t_0$. If the sum of the calculation time and the cross layer processing delay in P is T_{cal} before the reply is sent back, V will get the reply at $t_0 + \frac{2d_{PV}}{v} + \frac{L'+L}{w} + T_{cal}$. We find that the estimation of T_{cal} will impact both the upper bound and lower bound of the distance.

T_{cal} is impacted by various factors such as the CPU speed of the sensor, available memory, and the number of events to be processed before the current task. If the shortest processing time T_{min} and the longest latency T_{max} can be pre-determined, we will have:

$$\begin{aligned} &\frac{(T - (L + L')/w - T_{max}) \cdot v}{2} \leq d_{PV} \\ &\leq \frac{(T - (L + L')/w - T_{min}) \cdot v}{2} \end{aligned} \tag{4.5}$$

The computation time in P is easy to estimate. We assume that the sensors adopt a 8-round RC5 [108, 109] encryption method, whose security analysis can be found in [110]. RC5 has been combined with TinyOS [111] in several projects [112, 113]. The experiments show that it takes an iPAQ 3670 PDA 8 μs to decrypt a 128-bit data block. The computation efficiency of the hash function on a similar device has been studied in [22], which shows that 220K times hash calculation can be accomplished in one second. Therefore, the calculation in P can be accomplished within 20 μs.

The cross layer processing delay depends on the workload of the sensor and it is more difficult to estimate. To restrict the impacts of T_{cal} on the distance estimation accuracy, we adopt an alternative approach. Every $Prover\ P$ will start a local timer when it receives the last bit of the challenge. If the challenge has stayed in P for a time longer than T_{hold} and the reply has not been sent back yet, P will discard the reply and cancel the distance estimation. The $Verifier\ V$ will try this procedure later. In this way we can choose a suitable value of T_{hold} so that the error of d_{PV} can be pre-determined and both the upper and lower bound of the distance can be derived. In our experiments, we set T_{hold} to 2 ms so that the error of the measured distance is shorter than $3m$.

Local reconstruction

After measuring the distances to its neighbors, a sensor will broadcast these values in a packet. The packet will include four parts: the sender's identity, the neighbors' identities, the distances, and the keyed hash values to protect the integrity of the information. For example, a sensor s will broadcast (s, (s_1,$\overline{d_{ss_1}}$), $\cdots$, (s_q, $\overline{d_{ss_q}}$), $h_{K_{ss_1}}$(s, (s_1, $\overline{d_{ss_1}}$), $\cdots$, (s_q, $\overline{d_{ss_q}}$)), $\cdots$, $h_{K_{ss_q}}$(s, (s_1, $\overline{d_{ss_1}}$), $\cdots$, (s_q, $\overline{d_{ss_q}}$))), where s_1 to s_q are the neighbors.

After receiving the distance reports from its neighbors, a sensor will become aware of the network topology within two hops and it can reconstruct the local network using MDS. For those connections whose measurements from both ends are available, s will calculate the average value and put the result at the suitable positions in the distance matrix. If the measured distance is larger than the communication range, this neighbor relation will be aborted to avoid wormholes. The distance from a sensor to itself is 0. After the distances between the sensor pairs that can hear each other are calculated, a classical shortest-path algorithm, such as Dijkstra's algorithm, can be used to calculate the shortest distance between every sensor pair. When all positions in the distance matrix have been filled, MDS can be used to rebuild the network and a virtual position for every sensor will be generated. The local reconstruction results under two different scenarios (attack-free and under attack) are illustrated in Figure 4.3d and 4.3e.

Detection of wormhole

In this part, we first explain the distortions in the reconstructed networks that are caused by the wormholes from a mathematical view. Experimental results are then presented to support the analysis and to illustrate how the distortions can be used to locate the wormhole and identify the fake neighbor connections.

$$\begin{bmatrix} 0 & d_{1-2} & d_{1-3} & \cdots & d_{1-n} \\ d_{2-1} & 0 & d_{2-3} & \cdots & d_{2-n} \\ d_{3-1} & d_{3-2} & 0 & \cdots & d_{3-n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ d_{n-1} & d_{n-2} & d_{n-3} & \cdots & 0 \end{bmatrix} \longrightarrow \begin{bmatrix} b_{1i1} & b_{1i2} & \cdots & b_{1in} \\ b_{2i1} & b_{2i2} & \cdots & b_{2in} \\ \cdots & \cdots & \cdots & \cdots \\ b_{ni1} & b_{ni2} & \cdots & b_{nin} \end{bmatrix}$$

Figure 4.8. Generate B_i from the distance matrix D

If we have n sensors in the network, we can generate a $n \times n$ distance matrix D, as shown in Figure 4.8, in which every item d_{j-k} represents the distance between sensor j and k. If we choose sensor i as the origin of a space, we can build a $(n-1) \times (n-1)$ matrix B_i in which every item b_{jik} is determined as:

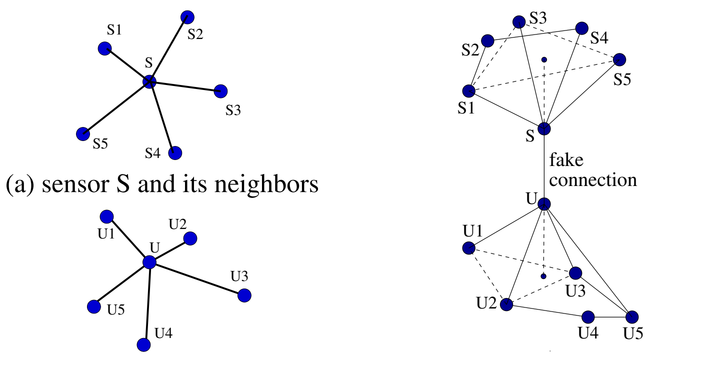
$$b_{jik} = \frac{1}{2}(d_{i-j}^2 + d_{i-k}^2 - d_{j-k}^2) \qquad (j, k \neq i) \tag{4.6}$$

This value can be viewed as the scalar product of the vectors $\overrightarrow{ij}$ and $\overrightarrow{ik}$. Since the three sensors i, j, and k will form a triangle, we can apply the cosine law and get the following result.

$$b_{jik} = d_{i-j} d_{i-k} \cos\theta_{jik} \tag{4.7}$$

Young and Householder [114] have shown that this matrix B_i can be factored as $B_i = X_i X_i^T$, in which X_i is the coordinate matrix of the sensors in the space with sensor i as the origin. The original analysis assumes that the distances among the sensors are accurate. For the scenarios in which the distances are fallible, an error matrix can be added to achieve the least squares bias.

With this analysis, a simple example can be used to illustrate the impacts of a wormhole on the local network reconstruction. We assume two groups of sensors: sensor s and its real neighbors s_1 to s_5, and sensor u and its real neighbors u_1 to u_5. The two groups of sensors are far away from each other in the real network and their layouts are shown in Figure 4.9a and 4.9b. Now we assume that a wormhole attack is conducted and a fake neighbor connection has been established between s and u. We choose sensor s as the origin of the space and generate the matrix B_s to illustrate the impacts of the wormhole.



(a) sensor S and its neighbors

(b) sensor U and its neighbors

(c) local reconstruction

Figure 4.9. Distortions in local reconstruction

Let us consider the triangle constructed by sensors s, u, and s_1. Since su is the only fake neighbor connection generated by the wormhole, we can use the Dijkstra's method to calculate d_{s_1u} as $d_{s_1u} = d_{su} + d_{ss_1}$. Therefore, we have:

$$\begin{aligned} b_{s_1su} &= \frac{1}{2}(d_{ss_1}^2 + d_{su}^2 - d_{s_1u}^2) \\ &= \frac{1}{2}(d_{ss_1}^2 + d_{su}^2 - (d_{ss_1} + d_{su})^2) \\ &= -d_{ss_1}d_{su} \end{aligned} \tag{4.8}$$

Since b_{s_1su} should also be equal to $d_{ss_1}d_{su}\cos\theta_{s_1su}$, we find that to fit this value in the matrix B_s we need to have $\cos\theta_{s_1su} = -1$. The wormhole will move node s_1 to the extension line of us so that the angle between $\overrightarrow{ss_1}$ and $\overrightarrow{su}$ can be equal to π. Similar condition will happen to the other neighbors s_2 to s_5, and u_1 to u_5. Since the real neighbor connections among these sensors will try to preserve the original layout during reconstruction, the final reconstruction result will be the joint impact of these two factors, as shown in Figure 4.9c.

From this example, we find that the wormhole can be viewed as an extra force that will push the sensors away from their original positions, thus leading to the following distortions: the distances and angles between the neighbor sensors in the reconstructed network will be very different from the values in the real layout. A good estimation of the distances and angles in the real layout can be acquired using the measured distances from Section 4.4.2. Below we present the experimental results to show these distortions.

We adopt a grid placement of sensors as shown in Figure 4.3a. Four different scenarios in local reconstruction are considered: (1) no wormhole exists within two hops, (2) one wormhole exists, (3) two independent wormholes exist, (4) multiple fake neighbor connections through the same wormhole exist.

Table 4.1
Distortions in edge lengths

Scenarios	$\overline{diff}$ / r
no wormhole	5.3%
one wormhole	24.7%
two wormholes	24.3%
one wormhole multiple connections	25.6%

The distortions in edge lengths can be measured by the average difference between the estimated distances between neighbor sensors and the length of the connections in the reconstructed network. If the measured distance between two neighbor sensors j and k is $\overline{d_{jk}}$ and the length of the reconstructed connection is d'_{jk}, the average difference can be calculated as:

$$\overline{diff} = \frac{\sum ||\overline{d_{jk}} - d'_{jk}||}{m}, \tag{4.9}$$

where m is the total number of neighbor relations in the reconstructed network. The ratios between $\overline{diff}$ and the communication range r under different scenarios are shown in Table 4.1. We find that when no wormhole exists in the local network, the reconstruction result will preserve the distances between sensors very well. On the contrary, as soon as the wormhole is included, the average difference will have a sharp increase. We can use this increase to detect the existence of a wormhole in the reconstructed local network.

The distortions in angles can be used to locate the fake neighbor connections. In the example shown in Figure 4.9, the real neighbors of node s will be moved to the extension line of us, which will lead to the decrease of the angles between these sensors. For every such an angle, two values can be determined: (1) θ_M, which can be calculated based on the measured distances from Section 4.4.2; (2) θ_R, which can be acquired from the reconstructed network. The distortions in angles can be measured by the differences between these two groups of values.

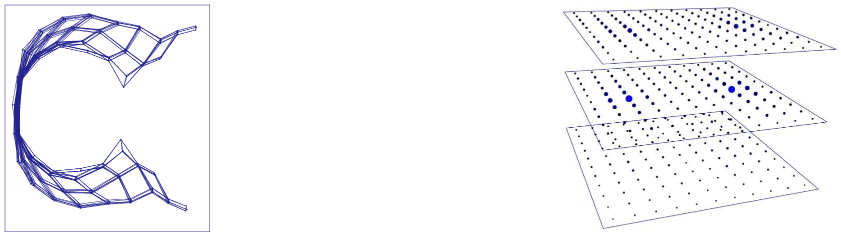
We define a new variable $wormhole\ indicator$ for every sensor i based on the differences in angles:

$$wormhole\ indicator(i) = \frac{\sum \theta_{diff-jik}}{q(q-1)};$$
$$\theta_{diff-jik} = \begin{cases} 0, & if\ \theta_{M-jik} - \theta_{R-jik} \leq \theta_{th}; \\ 1, & if\ \theta_{M-jik} - \theta_{R-jik} > \theta_{th}. \end{cases} \tag{4.10}$$

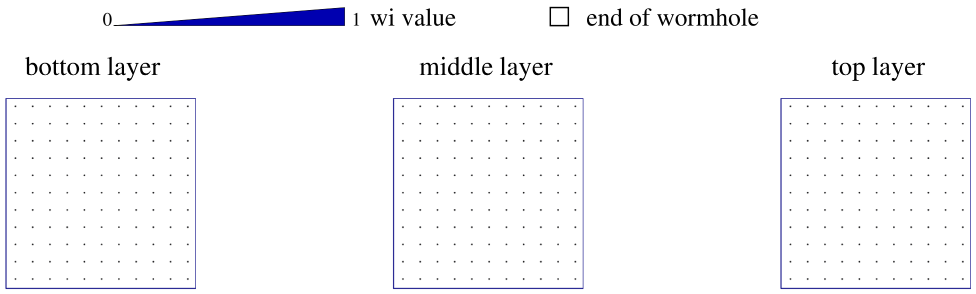
where i, j and k are neighbors, and q is the degree of connectivity of sensor i. From the definition we find that $wormhole\ indicator$ is a normalized variable with the value range [0,1]. Every sensor will calculate the wi value of itself and exchange it with the neighbors to locate the fake neighbor connections.

θ_{th} in equation 4.10 represents the threshold that is used to distinguish the changes in angles caused by the distance measurement inaccuracy from the distortions caused by the wormholes. In our simulation, θ_{th} has a format of $c \cdot \frac{vT_{hold}}{0.5r}$, in which vT_{hold} represents the distance measurement inaccuracy, r is the communication range, and c is a constant. When T_{hold} is not very long, $\frac{vT_{hold}}{0.5r}$ roughly describes the change in angles caused by the distance measurement inaccuracy. In our simulation, we set c = 4.

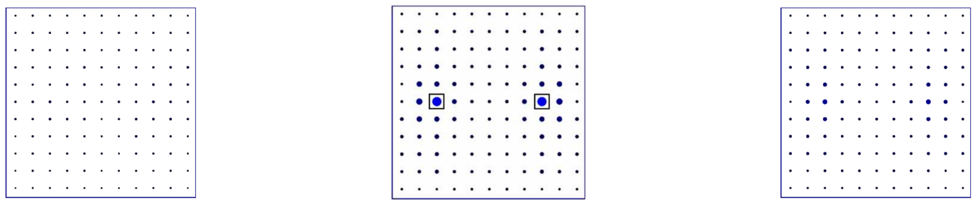
We have conducted extended simulation and the $wormhole\ indicator$ values under different scenarios are illustrated in Figure 4.10. For the clearance of the section, we show the wi values of the sensors in different layers separately. Every illustrated wi value is calculated by the sensor itself. From the figures, we find that the ends of the wormholes have the largest distortions in angles and they can be easily identified. In the following



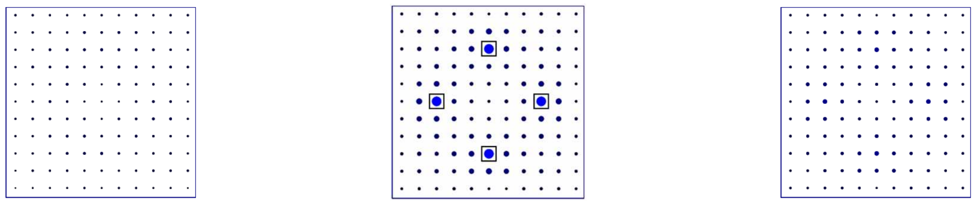
(a) Global reconstruction result and the view of $wormhole$ $indicator$ values in a 3D space when one wormhole exists.



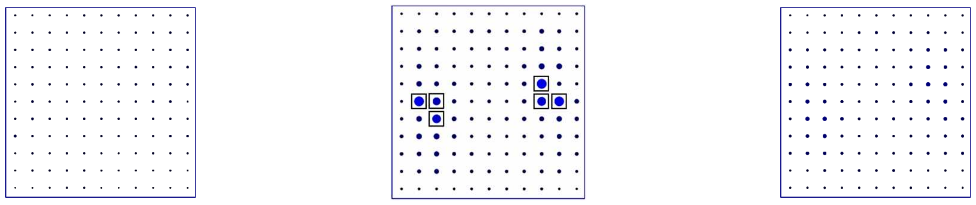
(b) $wormhole\ indicator$ values when no wormhole exists.



(c) $wormhole\ indicator$ values when one wormhole exists.



(d) $wormhole\ indicator$ values when two wormholes exist.



(e) $wormhole\ indicator$ values when nine fake neighbor connections go through the same wormhole.

Figure 4.10. $wormhole\ indicator$ values under different scenarios experiments, a neighbor connection will be labeled as a wormhole if both ends of the link have a $wormhole\ indicator$ larger than 0.3.

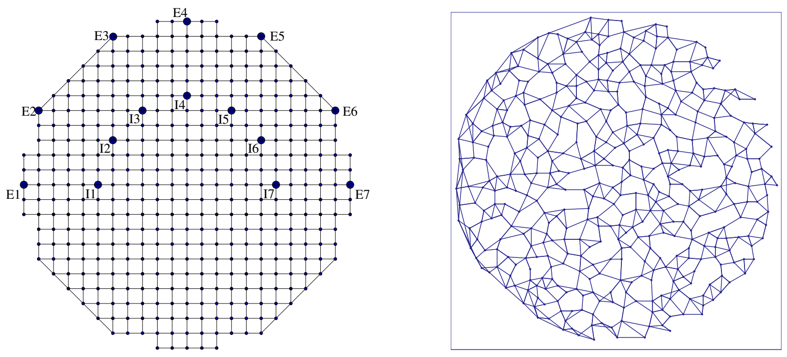
4.4.3 The Dis-VoW algorithm

With the readiness of all building blocks, we now walk through the steps of the Dis-VoW algorithm.

1. After deployment, every sensor will estimate the distances to the nodes that it can hear using the protocol described in Section 4.4.2.
2. Every sensor will broadcast the neighbor list and the distances so that its neighbors will be aware of the topology within two hops. The Dijkstra's method is used to calculate the shortest distance between every sensor pair and generate the distance matrix.
3. Using the classical metric MDS, every sensor will reconstruct the network within two hops and calculate a virtual position for every node in it.
4. Every sensor will calculate the $wormhole\ indicator$ value of itself. It will locate the wormholes as described in Section 4.4.2.
5. The detected wormholes will be avoided during routing discovery and packet forwarding.

4.5 Simulation results of MDS-VOW

The performance of MDS-VOW is examined through simulation. The experiments are conducted in two phases. In the first phase, we use $ns2$ to simulate the distance estimation procedures and the report of the information to the controller. The packets may get lost because of the unreliable medium. In the second phase, a Visual C++ program executes the MDS-VOW mechanism based on the received distances and tries to identify the fake neighbor connections. The sensors are deployed in a circle area instead of a square. This



(a)Grid placement (b)Random placement example

Figure 4.11. Experimental network topology

choice is inspired by the scenario that a security critical location is at the center of the circle, and we need to monitor the activities within a certain range. The area of the circle is $1km^2$, and the radius of the circle is about $565m$. The radio range r of the sensors is $110m$, and any two sensors that have a distance shorter than r can directly communicate to each other.

Two deployments of the sensors are examined: grid placement and random placement. In the grid placement, the sensors are deployed at an interval of $50m$ along the imaginary vertical or horizontal lines. A total number of 401 sensors are placed in the circle and the average degree of connectivity is 11.0. In the random placement, we apply the dart throwing method proposed in [115] to place the sensors randomly and roughly uniformly in the area. To maintain a similar degree of connectivity as in the grid placement, 441 sensors are used. Examples of the placements are shown in Figure 4.11. In both placements, the controller is located at the center of the circle. The justification of this sensor density choice is discussed in Section 4.7.

We model the distance estimation errors as uniform noises. If the accurate distance between two sensors is d $(d \leq r)$ and the error rate is e_m, a random value drawn from the uniform distribution $[d \times (1 - e_m), d \times (1 + e_m)]$ will be used as the measured distance. If the selected value is smaller than 0 or larger than the radio range, 0 or r will be used respectively. In the experiments, e_m changes from 0 to 0.8. For a fake neighbor connection through a wormhole, a random value from 0 to the radio range will be first selected as d, and then the error will be added. The data points in the figures represent the averages over 15 trials using different error values.

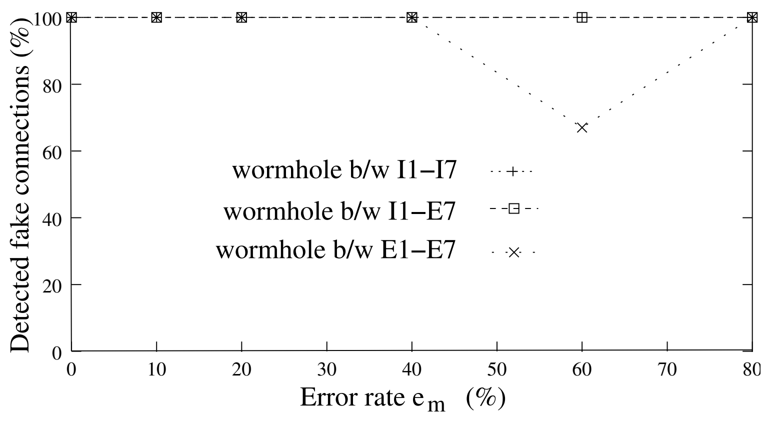
4.5.1 Grid placement

The grid placement of sensors is shown in Figure 4.11a. Seven sensors in the circle and seven sensors on the border of the area are selected as the potential victims of wormholes. They are labeled as $I1$ to $I7$ and $E1$ to $E7$ respectively. Two groups of experiments are conducted. The first group examine the MDS-VOW algorithm under different e_m rates. Four scenarios are considered: (1) no wormhole exists, (2) a wormhole links $I1$ and $I7$, (3) a wormhole links $E1$ and $E7$, and (4) a wormhole links $I1$ and $E7$. The detection accuracy of MDS-VOW and its impacts on routing are of special interest.

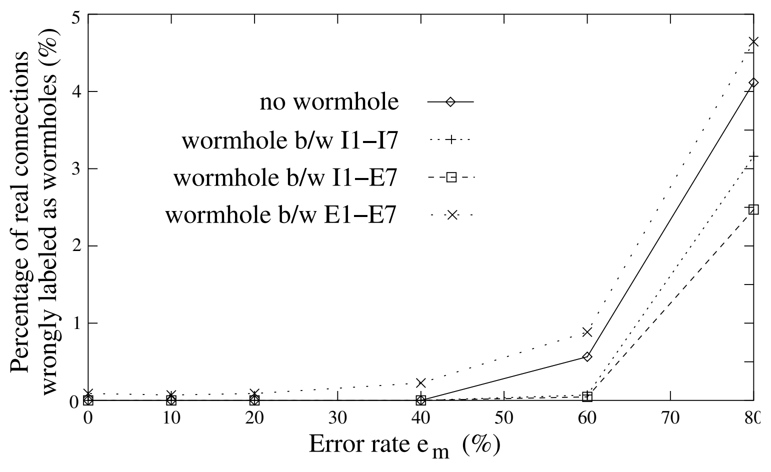
Figure 4.12 and 4.13 show the results. In Figure 4.12a, we find that MDS-VOW can detect the fake connections under most conditions when e_m increases from 0 to 0.8. MDS-VOW has a low false negative ratio. From Figure 4.12b we find that when e_m is smaller than or equal to 0.6, less than 1% of the real neighbor connections will be wrongly labeled as wormholes. When e_m increases to 0.8, the false positive alarm ratio becomes larger, but still less than 5% of the real connections are wrongly accused.

The false positive alarms will lead to the breaks of the real neighbor connections and the increase in the average path length. If all connections of a sensor are broken, an isolated node will be generated and the events detected by such sensor cannot be transferred out. To examine the impacts of the false positive alarms, we show in Figure 4.13 the increase in the average path length between all sensor pairs. Since the degree of connectivity in the original layout is relatively large, the increase in the average path length is small. We do not detect isolated sensors in the experiments.

In the second group of experiments, we fix the choice of e_m at 0.4 and examine the detection accuracy of MDS-VOW when the distance between the two ends of the wormhole changes. Seven sensors in the circle and seven on the border are selected as the potential



(a) detection accuracy of MDS-VOW



(b) false positive alarms of MDS-VOW

Figure 4.12. Detection accuracy of MDS-VOW in grid placement when the error rate e_m changes

victims. Since the increase in the average path length is small in Figure 4.13, we focus on the false alarm ratio in this group of experiments.

The ends of the wormhole are put at different positions in the network and three conditions of the fake connection are examined: (1) one end of the fake connection is $I1$, the other end changes from $I2$ to $I7$, (2) one end is $I1$, the other end changes from $E2$ to $E7$, and (3) one end is $E1$, the other end changes from $E2$ to $E7$. The results are shown in Figure 4.14 and under most conditions the fake connections can be effectively located and not many false positive alarms are introduced into the network.

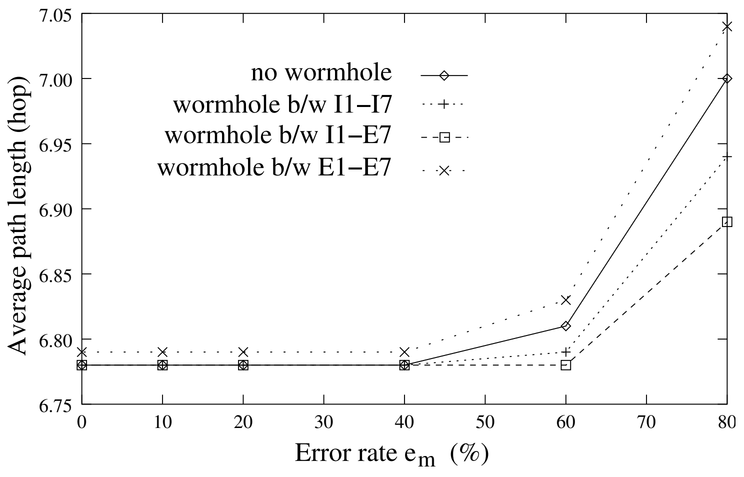
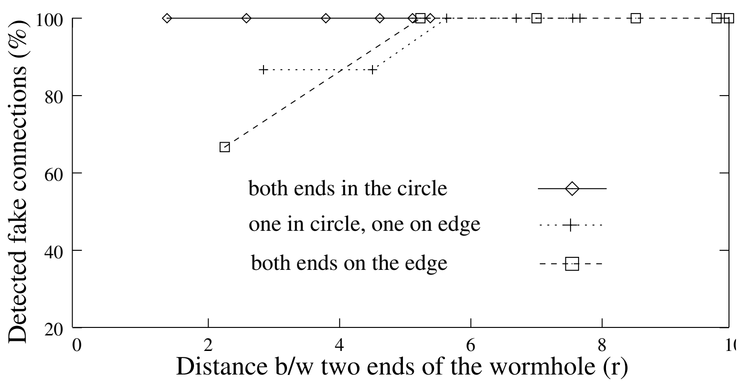
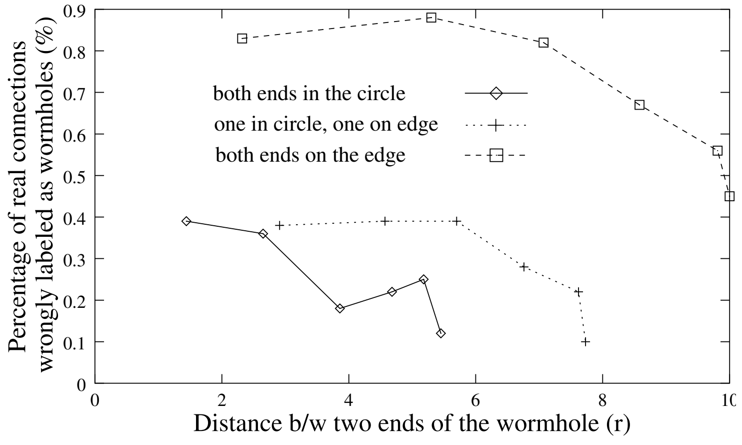


Figure 4.13. Increase in the average path length caused by false positive alarms

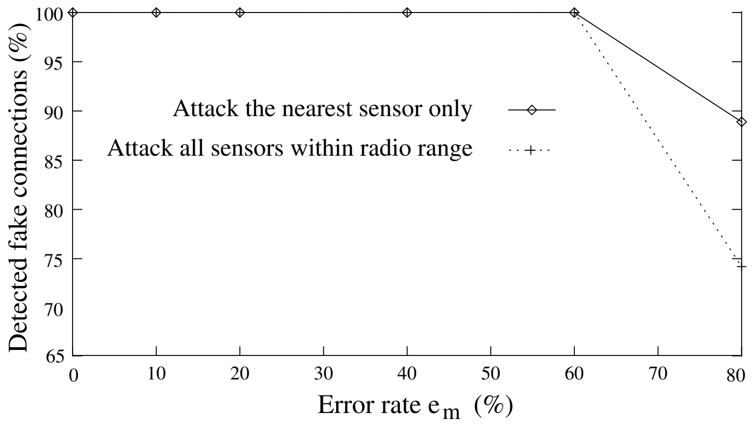


(a) detection accuracy of MDS-VOW

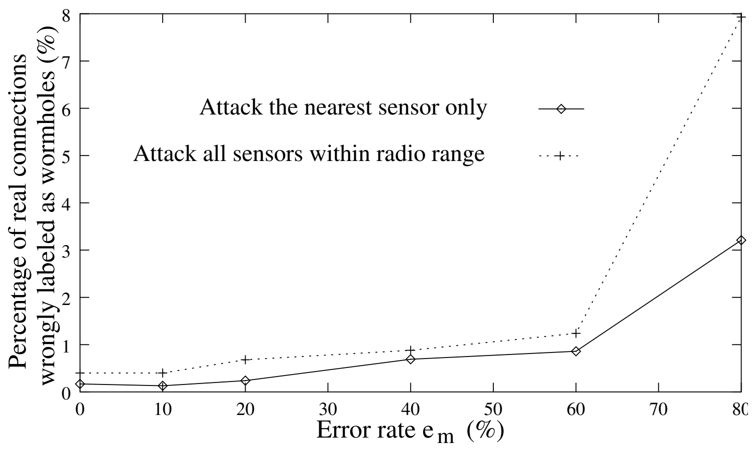


(b) false positive alarms of MDS-VOW

Figure 4.14. Detection accuracy of MDS-VOW when varying wormhole length



(a) detection accuracy of MDS-VOW



(b) false positive alarms of MDS-VOW

Figure 4.15. Detection accuracy of MDS-VOW in random placement

4.5.2 Random placement

In random placement scenarios, we apply the dart throwing method to place the sensors randomly and roughly uniformly in the network. One layout is shown in Figure 4.11b. Only a part of the neighbor connections are shown in the figure to assist the understanding of the topology. Two groups of experiments are conducted to examine the detection accuracy of MDS-VOW. In group one, two random positions in the area are selected as the ends of the wormhole. The wormhole then chooses a sensor from each end that has the shortest distance to it. If the distance between the two sensors is larger than r, a fake connection between them will be established. Otherwise, the position of the wormhole will be generated again. Experiments are conducted by varying the error rate e_m.

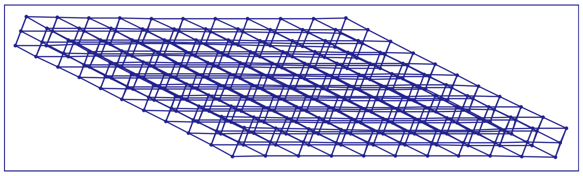
In the second group of experiments, there is still only one wormhole in the network. But the wormhole will establish fake connections between all sensor pairs when the sensors are within the radio range to the ends of the wormhole. For example, if s_1 to s_3 are the sensors within r to one end of the wormhole, and s_4 to s_6 are within r to the other end of the wormhole, 9 fake connections will be established if the distance between every pair is larger than r. Experiments are conducted by varying e_m and the results are illustrated in Figure 4.15.

Studying the results shown in Figure 4.12 to Figure 4.15, we find that when $e_m \leq 0.6$, MDS-VOW has a low false positive ratio and a low false negative ratio. The proposed mechanism can detect the fake connections in the grid placement and random placement scenarios without hurting many real connections.

4.6 Simulation results of Dis-VoW

The detection accuracy of Dis-VoW is studied through simulation using $ns2$. During the distance estimation procedures and the broadcast of the neighbor lists, the packets may get lost because of the unreliable medium. Every sensor will reconstruct the local network within two hops and locate the wormholes in it. The sensors are deployed in a three-dimensional space with the size of $700m \times 700m \times 140m$. This layout tries to simulate a scenario in which the sensors are deployed in a water area with a certain depth range. The communication range of the sensors is $150m$, and any two sensors having a distance shorter than this can directly communicate to each other.

Two deployments of sensors are examined: grid placement and random placement. In the grid placement, the sensors are deployed in three layers and the distance between two neighbor layers is $70m$. In each layer, 11×11 sensors are placed at an interval of $70m$ along the imaginary vertical or horizontal lines. A total number of 363 sensors are used and the average degree of connectivity is 18.4. In the random placement, we apply the dart throwing method proposed in [115] to place the sensors randomly and roughly uniformly in the 3D space. A total number of 256 sensors are deployed and a similar degree of connectivity is maintained. The examples of the two placements are shown in Figure 4.16a and 4.16b.


(a) Grid placement example: a $11 \times 11 \times 3$ grid.

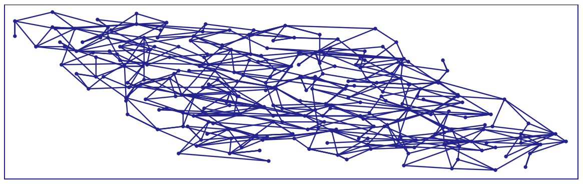
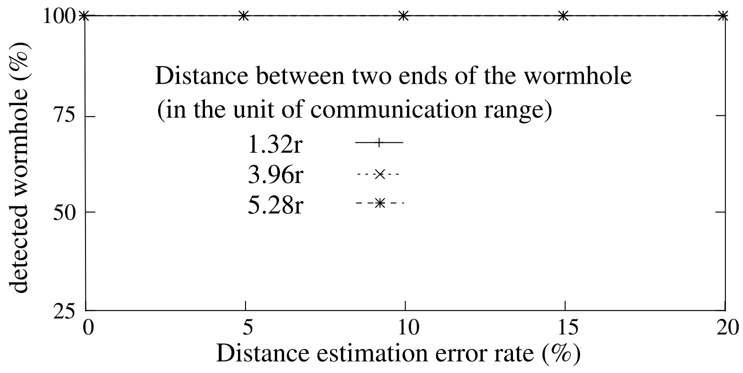

(b) Random placement example.

Figure 4.16. Experimental network topology

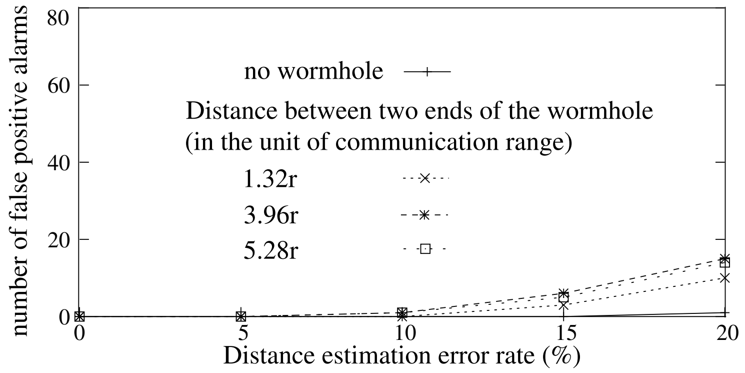
Since the sensors share an acoustic channel to communicate with each other, the distances to different neighbors of a node cannot be measured at the same moment. The differences in measurement time and the relative movement between sensors will introduce an inaccuracy into the network reconstruction. To study the impacts of this inaccuracy on the detection capability of Dis-VoW, we apply an error to the measured distances. If the real distance between two sensors is $d (d \leq r)$ and the error rate is e_m, a random value drawn from the uniform distribution $[d \times (1 - e_m), d \times (1 + e_m)]$ will be used as the measured value. Since the relative moving speed between sensors is slow, we examine different values of e_m from 0 to 0.2. The justification of this choice is discussed in Section 4.7. Every data point in the figures represents the average value over 10 trials using different error files.

4.6.1 Grid placement

In grid placement of sensors, two groups of experiments are conducted to examine the detection accuracy of the proposed mechanism. In the first group, only one wormhole exists in the network. Three pairs of sensors on the diagonal of the middle layer in the grid are selected as the potential victims of the wormholes. The real distances between the three pairs of sensors are 1.32, 3.96, and 5.28 times of the communication range. Different values of e_m from 0 to 0.2 are examined. The detection accuracy and the number of false positive alarms are of special interest.



(a) Detection accuracy with various distance estimation errors.

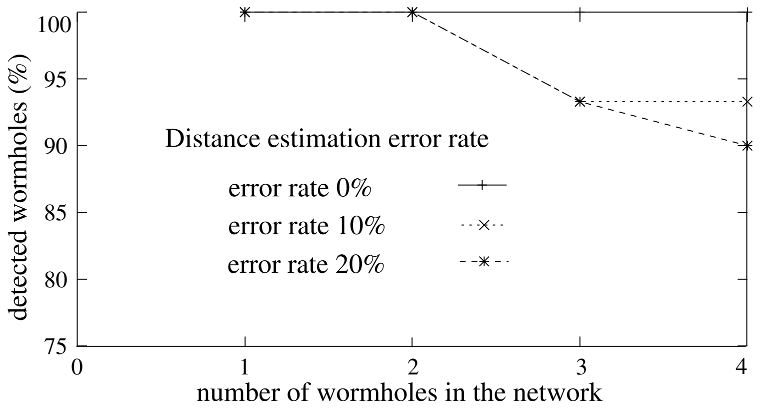


(b) Number of false positive alarms.

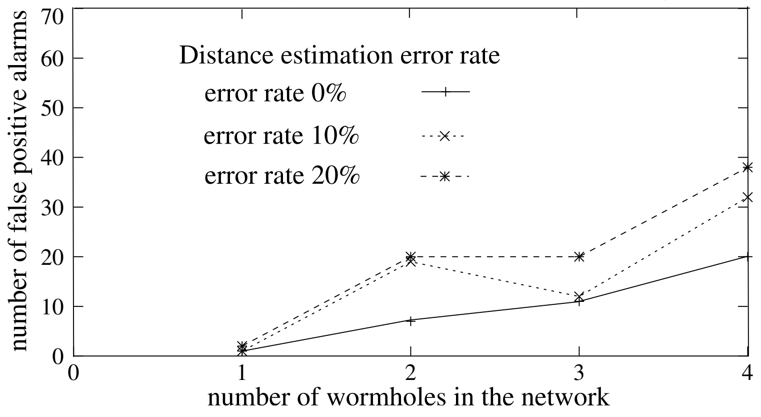
Figure 4.17. Detection accuracy of Dis-VoW with different error rate e_m

Figure 4.17 shows the results. In Figure 4.17a, we find that Dis-VoW can detect the fake neighbor connections under most conditions when there is only one wormhole in the network. The proposed mechanism is robust against the distance estimation errors when e_m is not larger than 0.2. Figure 4.17b shows the number of false positive alarms introduced by Dis-VoW. Compared to the results in [116], we find that Dis-VoW has a lower false positive alarm rate than MDS-VOW. The main reason is that the new method is a distributed approach. Therefore, if the wormhole is not within two hops to a sensor, its local reconstruction will not be affected and no false alarm will be caused.

In the second group of experiment, we examine the detection accuracy of Dis-VoW when the number of wormholes in the network changes. The victims of every wormhole are randomly and independently selected from the sensors as long as the distance between them is longer than the communication range.



(a) Detection accuracy when number of wormholes changes.



(b) Number of false positive alarms.

Figure 4.18. Detection accuracy of Dis-VoW with different numbers of wormholes

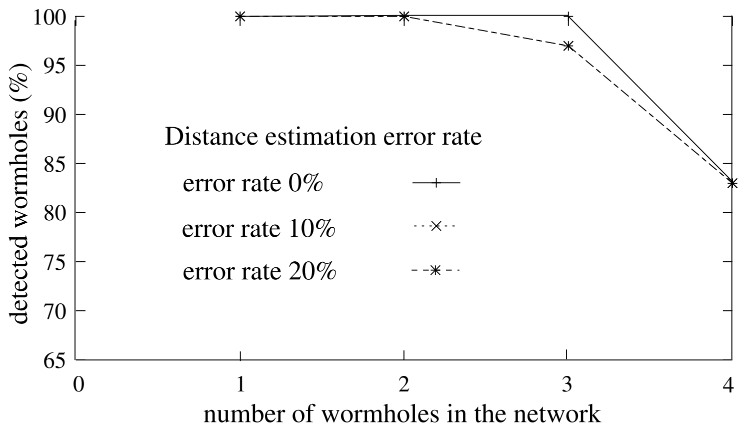
The results are illustrated in Figure 4.18. Compared to the values in Figure 4.17, we find that when the number of wormholes increases, the detection accuracy will decrease and more false positive alarms will be introduced into the system. The main reason that leads to the decrease of the performance is as follows: although the victims of the wormholes are randomly and independently selected, as the number of wormholes increases, the probability will also increase that they are close to each other and multiple wormholes will impact the local reconstruction jointly. The co-existence of multiple wormholes in a small area will lead to more complex distortions. Multiple rounds of detection may be required to deal with such conditions and more details will be discussed in Section 4.7.

4.6.2 Random placement

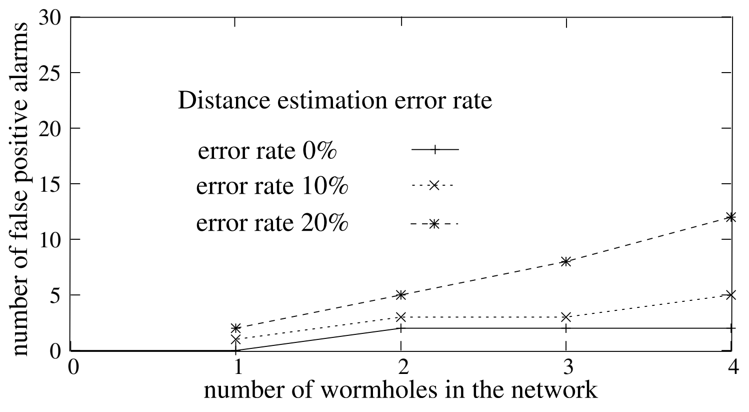
One example of random placement of sensors is illustrated in Figure 4.16b. To maintain a similar degree of connectivity as in the grid placement, we apply the dart throwing method [115] to deploy the nodes and we require that the distance between any two neighbor sensors will not be shorter than $60m$. In the illustrated example, a total number of 256 sensors are placed and the average degree of connectivity is 18.3.

Since the simulation results in grid placement show that Dis-VoW can detect most of the fake neighbor connections when there is only one wormhole in the network, in the random placement we focus on the scenarios in which multiple wormholes exist. The victims are randomly and independently selected from the sensors. The results are shown in Figure 4.19. We find that the curves in Figure 4.19 are very similar to those in Figure 4.18. The results show that the deployment of sensors does not impact the detection accuracy of Dis-VoW to a large extent.

Studying the results illustrated in Figure 4.17 to Figure 4.19 for different sensor deployments, we find that the distributed approach can identify most of the fake neighbor connections when there is only one wormhole in the network. The proposed mechanism is robust against the distance estimation errors. When there are multiple wormholes in the



(a) Detection accuracy when number of wormholes changes.



(b) Number of false positive alarms.

Figure 4.19. Detection accuracy of Dis-VoW in random placement of sensors

network, the approach can still provide a decent detection accuracy without introducing many false positive alarms.

4.7 Discussion

The proposed mechanisms detect wormholes by visualizing the anomalies caused by such attacks in the reconstructed network. They avoid the requirements of special hardware and can be applied to the environments such as sensor networks. The mechanisms consist of multiple steps and each step uses the result from the previous one as input. The details of the method for each step are transparent to other steps and they can be improved independently.

4.7.1 Impacts of sensor density

The sensor density impacts the algorithms in two ways: (1) The calculation of the distance matrix, and (2) The performance degradation caused by false positive alarms. Assuming a network with infinite sensor density. When two sensors h_1 and h_2 have a distance d, where $r < d < 2r$, we can always find a third sensor h_3 that is on the line determined by h_1 and h_2 and has a distance shorter than r to both of them. Therefore, when using the Dijkstra's method, we can add $|h_1h_3|$ and $|h_2h_3|$ to calculate $|h_1h_2|$ without introducing any error. As the density decreases, errors will be introduced into the distance matrix even when the distance estimations between neighbor sensors are accurate. For the same reason, when sensor density decreases, the degree of connectivity becomes smaller, and the node has a higher probability to become an isolated sensor when the false positive alarms break the real connections.

We refer to previous research efforts and experiments in real applications when choosing the sensor density in our experiments. Similar density has been adopted in [83]. In that paper the authors deploy 200 nodes in a $10l \times 10l$ area when the radio range changes from l to $2l$. In the vehicle classification experiments conducted by U.S. Army [117], the sensors are deployed at an interval of $30 - 70m$.

4.7.2 Safety of the proposed mechanisms

As security enhancements to sensor networks to defend against wormhole attacks, the robustness of the proposed mechanisms must be studied to avoid introducing new vulnerabilities. During the execution of the protocols, the malicious nodes can conduct the attack by: (1) changing the contents of the packets or impersonating the senders when retransmitting them, (2) discarding these packets, and (3) misleading the distance estimation. Now we discuss the solutions to these attacks respectively.

For the first attack, the integrity of information can be protected by the secret keys shared by the sensors. The latest approaches for key distribution in sensor networks have been discussed in Section 4.2. For the second attack, the protocols can require that any sensor who fails to accomplish the distance estimation procedure or the exchange of neighbor lists will not be considered as a real neighbor. In this way, the malicious nodes cannot discard these packets to hide the fake neighbor connections. The malicious nodes can still control the retransmission power or the buffering time of the packets to mislead the distance estimation procedures. But since the communication range is known to the sensors, the impacts are restricted and it can be viewed as a special distance estimation error.

4.7.3 Control of false positive alarms

During the detection procedures, the false positive alarms will lead to the increases in the average path length and the end-to-end delay between sensors. In the worst case, isolated nodes will be generated. Therefore, false alarms must be controlled.

An extra step can be adopted by MDS-VOW and Dis-VoW to reduce the false positive alarms. With all the detected fake neighbor connections (could include some false alarms) excluded, a second round of reconstruction can be conducted. The rebuilt network will be very similar to the real layout of sensors and we can determine whether a "detected wormhole" is a false alarm by examining the distance between the sensor pair. This method will add the excluded connections back to the network one-by-one so that the real wormholes leading to the distortions can be located. This method will double the computation overhead to improve the detection accuracy and it is extremely helpful in the environments when the degree of connectivity is not large.

4.7.4 Frequency to conduct wormhole detection

The relative movement between underwater sensors will lead to the changes in network topology, which will put two challenges to Dis-VoW: (1) Since the distances to different neighbors cannot be measured at the same moment, the values used in local reconstruction can be inaccurate. The impacts of this inaccuracy must be studied. (2) Since wormholes can be formed dynamically when the connections among sensors change, wormhole detection must be conducted repeatedly during the network lifetime. Below we discuss these problems respectively.

For the first problem, we find that for most sensors, the distance measurement inaccuracy introduced by sensor movement during the distance estimation procedures and the collection of neighbor lists is shorter than $8m$. If the lengths of the neighbor connections are uniformly distributed from 0 to the communication range, this inaccuracy will introduce a 10% distance estimation error on average. This analysis leads to our choice of e_m from 0 to 20% during simulation.

For the second problem, wormhole detections can be conducted in a proactive or a reactive method. In the proactive approach, the sensors will choose an interval T_i and conduct wormhole detections every T_i seconds. This interval determines the longest time that a wormhole can exist before it is detected. Compared to this scheme, the reactive method is more efficient. Since a wormhole can be formed dynamically only when a neighbor relation between two nodes changes, a sensor can estimate the time that the next change happens based on the distances and the relative moving speed and conduct the detection reactively.

4.8 Conclusion

Different from the previous approaches, MDS-VOW and Dis-VoW defend against wormhole attacks in sensor networks without depending on any special hardware. They use the inaccurate distance estimations among sensors as inputs, and rebuild the network layout using multi-dimensional scaling. The analysis and experiments show that the wormholes distort the reconstructed network to fit the fake neighbor connections. A normalized variable $wormhole\ indicator$ is defined based on these distortions to locate the wormholes. MDS-VOW and Dis-VoW consist of multiple steps and each step can be improved independently.

Experimental studies using grid placement and random placement of sensors are conducted to examine the detection accuracy of the proposed mechanisms. The results show that they can detect most of the fake neighbor connections without introducing many false positive alarms when the distance measurement errors are not large. As a distributed approach, Dis-VoW will introduce a limited amount of computation and storage overhead to the sensors. The impacts of sensor density, the security of the proposed mechanisms, the method to reduce false positive alarms, and the frequency to conduct wormhole detection are discussed to provide a more comprehensive view of the methods.

Additional research is under construction. We plan to apply the distributed detection mechanism to land-based sensor networks in 3D environments. Experiments that study the joint impacts of multiple wormholes on local network reconstruction are being conducted. The research will lead to a more accurate and efficient solution that can defend against wormhole attacks in various scenarios.

5 SECURITY STUDY OF AD HOC ROUTING PROTOCOLS

5.1 Introduction

The limited power resource and computation capabilities of mobile devices determine their heavy dependence on other nodes for data accessing and information processing. A reliable network topology must be assured through efficient and secure routing protocols for mobile ad hoc networks to enable the pervasive computing.

Many efficient routing protocols for ad hoc networks have been proposed. We may classify them by the time that the routing information is acquired. In the on-demand (reactive) protocols, such as AODV [5], Dynamic Source Routing (DSR) [7], and Temporally Ordered Routing Algorithm (TORA) [118], the routing information is required and maintained only when it is needed. In the proactive protocols, such as Destination Sequence Distance Vector (DSDV) [4], and Clusterhead Gateway Switch Routing (CGSR) [26], the mobile nodes exchange information routinely and construct the routing tables in advance. There are other protocols, such as Zone-based Routing Protocol (ZRP) [119], that employ both mechanisms.

The original versions of the protocols do not consider much on security and robustness. But the routing topology of ad hoc networks is prone to both external and internal attacks in the application environments such as battlefields. Research has been carried out to protect mobile ad hoc networks. The adopted mechanisms include: providing decentralized public key infrastructure [17, 70], distributed monitoring and evaluating node behaviors [8, 9], and using hash chain and digital signature to guarantee the integrity of the information [10–12]. These methods protect the ad hoc networks from some attacks. However, they face the following difficulties:

(1) The restrictions on power consumption and computation capabilities prevent the usage of complex encryption algorithms. The time synchronization cannot be efficiently

achieved for hash chains. (2) The constantly changing topology and dynamic membership increases the difficulty of authentication and key distribution. (3) Some attacks cannot be detected by the localized monitoring. Therefore, intrusion detection and intruder identification based on these methods are restricted.

Research is required to ascertain the potential connections between the essential properties of the routing procedures and the security vulnerabilities introduced by them. Then security enhancements addressing these deficiencies can be designed efficiently. This research provides a detailed analysis on security properties of two representative ad hoc routing protocols, namely, AODV and DSDV. We especially examine the difference caused by on-demand and proactive mechanisms. Many examined properties, such as distance vector, and destination sequence, are also adopted by other ad hoc routing protocols. Thus the results can be applied beyond AODV or DSDV and provide guidelines for the design of a secure routing protocol and the Intrusion Detection Systems (IDS) for ad hoc networks.

The remainder of the chapter is organized as follows: Section 5.2 presents the related work. Section 5.3 exploits some attacks on the protocols and compares the security deficiencies caused by on-demand route query and the proactive mechanism. Section 5.4 illustrates the damages of false distance vector attacks and false destination sequence attacks by simulation. Section 5.5 presents the anomalous patterns of sequence numbers that can be used to detect false destination sequence attacks. Section 5.6 concludes the chapter.

5.2 Related work

There are efforts, in both theory analysis and project development, to investigate the security of ad hoc networks, to establish IDS, and to construct secure communication protocols.

Zhang and Lee presented a generic multi-layer integrated IDS structure [15]. But how to efficiently collect the patterns of attacks and how to safely distribute the intrusion detection results to other nodes are not discussed in detail. Bhargavan, Zhou and Haas explored the security issues of wireless LANs and ad hoc networks [17] [16]. They summarized the primary questions to achieve security and the challenges to the routing protocols.

Providing decentralized public key infrastructure is a fundamental problem in securing ad hoc networks. Hubaux and his colleagues proposed a key distribution mechanism similar to PGP [70, 120]. They present a practical solution to the key management problem stated by Haas in [17]. But the transfer of trust among mobile nodes is difficult to apply under some critical environments.

Distributed monitoring and evaluating node behaviors is also popular in security enhancements. A system integrating watchdog and pathrater with DSR is presented in [8]. AODV-S [9] enables the neighbors to collaboratively authorize a token to the node before it can join the activities in the network. But there are attacks that cannot be detected locally and have long delays before the anomaly is discovered. They put challenges on secure information storage and sharing.

Several protocols using hash chain, digital signature or both to guarantee the integrity of routing information have been proposed. [10] uses both mechanisms to protect the routing procedure. SEAD [11] uses one-way hash chains to provide authentication. Ariadne [12] uses a variant of TESLA to achieve similar goals. These protocols may not suffer the attacks exploited in Section 5.3, but the synchronization among mobile devices is not easy to achieve. The evaluation of secure routing in ad hoc networks can be found in [121]. More secure protocols and IDS structures can be found in [14, 18, 33, 122].

5.3 Attack analysis and security comparison

We exploit some attacks on AODV and DSDV to expose the potential linkage between the essential features of the protocols and their security flaws. The primary difference between AODV and DSDV is that they work in on-demand and proactive modes separately. It leads to the difference in the conduction costs of attacks, propagation procedures of false routes, and the detection of attacks.

5.3.1 Classification of attacks

We first divide the attacks into passive and active categories. At a finer level, we group the active attacks by their target features.

Passive attacks

A malicious node conducts a passive attack by ignoring operations supposed to be accomplished by it. One example of passive attacks on AODV or DSDV is silent discard, carried on by an intermediate node along the forwarding path. Instead of forwarding a packet to the next hop, the attacker drops the data silently. Another example is partial routing information hiding. It is conducted by a malicious node in DSDV by hiding the available paths to specific nodes when it broadcasts its routing table, or in AODV by ignoring to give out RREP when an active route is available.

It is usually difficult to distinguish passive attacks from Byzantine failures [123] in ad hoc networks. For example, a packet drop can also occur because of node movement or unreliable wireless medium. Fortunately, the constantly changing topology and multiple available paths among wireless nodes limit the impacts of passive attacks. For example, our simulation shows that [124] in an ad hoc network that has 30 nodes and 25 connections, the silent discard by one malicious node may cause the delivery ratio to decrease 3%. We do not put more efforts on the analysis of passive attacks because they rely more on the network topology than the protocol characteristics.

Active attacks

The malicious node generates an active attack by introducing false information into an ad hoc network. It confuses routing procedures and degrades network performance. In DSDV the false information is carried in the routing packets. In AODV, the RREP is especially attractive to attackers because the reverse routes established by RREQ will become expired in a short time if no active traffic uses those routes. Two active attacks that threat both AODV and DSDV are:

- False distance vector attack

In both AODV and DSDV the mobile nodes collect routing information solely from direct neighbors. The incomplete understanding of global topology enables the false distance vector attacks. The malicious node can claim that the destination is one (or a few) hop(s) from it in the routing update packets or RREP even if it does not have any available path in its routing table. If no other replies provide a fresher or shorter route, the source will choose the path provided by the malicious node, and the data packets will be dropped or compromised.

- False destination sequence attack

Both AODV and DSDV employ destination sequence to identify the freshness of routing information. When multiple routes are available, the source always chooses the one with the largest sequence number. By assigning a large false destination sequence in the routing update packets or RREP, the attacker's reply can easily beat other replies and attracts the data traffic. Even worse, the deceived nodes will propagate in good faith the false route to other nodes, thus strengthening the impacts of the attack.

5.3.2 Security analysis

Security comparison

The primary difference between AODV and DSDV is the adoption of on-demand and proactive methods separately. Each of the methods brings advantages and disadvantages in security. While the on-demand route query enables low protocol overhead and adaptability to node movement, it also leaves a lenient space to the attackers. In proactive protocols the malicious node can send multiple false routes in the same packet. The detailed comparison on security comes as follows:

The on-demand property enables the malicious nodes to conduct real time attacks. Most of the attacks on AODV do not need any preparation or establishment time. For example, when a malicious node receives a RREQ, it can immediately form a false route reply and conducts the attack. As a comparison, when the malicious node attacks a proactive protocol, it must send out the false information in advance and has to routinely update the fake route to keep it alive. The longer a false route exists, the larger probability that it is detected. At this point, it is difficult to catch an on-going attack on a reactive protocol before it causes performance degradation.

The on-demand property enables the attackers to make flexible choices on the targets, the methods, and the points in time of attacks. For example, the malicious node can choose to attack all connections to or from a specific node. It can attack the same node with different methods. As to one victim, the attacker can choose to send false replies to some of the route queries while leaving others untouched. As a comparison, an attack on a proactive protocol usually does not have the flexibility. For example, a false route with a large sequence will be propagated to all other nodes through route exchanges. It is difficult for the malicious node to attack a specific connection without impacting others. This stiffness increases the probability that the attacker is detected and located.

It is more difficult to trace back the sources of false information in AODV. The routing reply is unicasted back to the source. Unless the mobile nodes monitor all nearby traffic, there will be only one node along the false route that directly receives the false information from the attacker. For the intruder identification algorithms that use quorum voting to locate the attacker, AODV is less efficient on the trace back procedures.

Communication overhead of attacks

The communication overhead caused by sending false routes in AODV is determined by the width and frequency of attacks. For example, if the malicious node wants to attack one specific connection, it only needs to send a single false RREP. As the other extreme condition, if the malicious node wants to attack every connection to every other node, it has to send many false RREP. In DSDV, the overhead is more consistent. The attacker can send many false routes in the same routing packet. At this point, attacking proactive protocols is more communication efficient for an aggressive attacker.

Propagation of false routes

In AODV, the false RREP will be unicasted back to the source. In [125] it has shown that the average path length is proportional to the square root of node density in ad hoc networks. Therefore the number of nodes cheated by a false RREP is proportional to that order. Because an intermediate node may send out RREP to other route queries afterwards, the false routes will form a tree rooted at the malicious node. In a proactive protocol, the false routes will be transmitted within a growing round area by the routing exchanges. At this point, a single false route in AODV propagates slower and has weaker impacts.

Cancelation of false routes

As the IDS in ad hoc networks develop, the malicious node sometimes has to cancel the false routes originated from it to avoid being identified. In most ad hoc routing protocols, the updates to current routes are caused either by the break of an active link or the appearance of a fresher or shorter path. The attacker in DSDV can stop sending false routes to cancel the impacts. The new updates will be propagated to the neighbors and the false routes will be smoothly replaced. The number of nodes that notice this change depends on the propagation range of the false routes. In AODV, when the attacker stops sending packets, the neighbors will assume that the link is broken. The re-discovery procedure will broadcast RREQ. At this point, it is more difficult for the attackers in AODV to silently cancel the impacts of false routes.

Detection of false routes

It is difficult to detect false distance vector attacks in AODV and DSDV because the nodes cannot construct the global view of the connectivity. The false destination sequence attacks can be detected by the victim if it finds that the sequence has never been generated by it. Because in DSDV the nodes routinely exchange their routing tables, we can estimate the maximum propagation delay of the false sequence from the attacker to the victim by the product of routing packet broadcast interval and their distance in hops. If the false sequence outruns the real number when it arrives at the victim, the attack will be detected. In AODV, the false sequence can be detected only when the false path is broken and the re-discovery procedure broadcasts a RREQ carrying the false number. It depends on the mobility of the nodes and no upper limit can be predicted. More details about the detection of false sequence attacks will be discussed in Section 5.5.

5.4 Simulation results

We study the practical impacts of the attacks and examine our analysis through simulation. Two attacks on AODV and DSDV are considered: false distance vector and false destination sequence. Except sending false routes, the attacker will discard any data packets passing through it. Two traffic conditions are tested. Under condition one, all connections have the same destination. This condition is chosen to simulate the scenario in which the malicious node only attacks the hot point in the applications and tries to block traffic to it (e.g. block soldier's reports to officer, or sensor's reports to information-sink). We measure the delivery ratio, attack overhead, and the propagation of false routes when the malicious node sends false routes about the common destination. Under condition two, a more sophisticated traffic scenario is used. We study the delivery ratio and attack overhead against the mobility of the wireless nodes.

The simulation of attacks is deployed using ns2 [6]. Table 5.1 lists the simulation parameters that we use.

The choices of the parameters consider both accuracy and efficiency of the simulation. The node moving speed covers a range from human jogging to vehicle riding in country field. Faster speed is not considered because the frequency of route changes will confuse the performance degradation caused by attacks. The packet rate is chosen to avoid congestion even when there are multiple connections converging at the same node.

Table 5.1
Simulation parameters

Simulator	ns-2
Examined protocols	AODV, DSDV
Simulated attacks	False distance vector, False destination sequence
Simulation duration	1000 seconds
Simulation area	1000 * 1000 m
Number of mobile nodes	30
Transmission range	250 m
Movement model	Random waypoint
Maximum speed	5 – 20 m / s
Traffic type	CBR (UDP)
Data payload	512 bytes
Packet rate	2 pkt / s
Number of malicious node	1
Pause time	10 seconds

We choose the following metrics to evaluate the impacts of attacks: (1) packet delivery ratio, (2) false routing packets sent by the attacker, (3) the number of innocent nodes that are cheated by the false routes.

Metric (1) is selected to evaluate the percentage of packets that are affected by the attacks. This can be viewed as the “strength” of an attack. Metric (2) is used to examine the communication overhead of different attacks. Metric (3) examines the propagation of false routes and the potential impacts that are not shown by metric 1. Combining metric 2 and 3, we can examine the efficiency of the attacks.

5.4.1 Simulation condition one

Under condition one, all connections have different sources and use node 29 as the destination. Node 5 is the malicious node. In AODV, it sends false RREP to every RREQ that it receives. In DSDV, it sends false routing information about node 29 in the route update packets. We study the selected parameters against the number of connections. Because there are thirty nodes in the network, the maximum number of connections from different sources to node 29 is twenty-eight (except node 5 and 29). The maximum speed of node movement is 5m/s. Every point in the figures is the average value of ten simulation scenarios. To calculate the number of nodes getting cheated by the false routes, the routing trees to node 29 are examined every 50 seconds. Figure 5.1 to 5.4 show the results.

Figure 5.1 shows the delivery ratio versus the number of connections to node 29 under three conditions in both protocols: when node 5 does not conduct attacks, when it attacks the routes with false distance vector, and when it attacks with false destination sequence. It is easy to tell that the impact of false destination sequence attack on delivery ratio is much more severe than that of false distance vector attack. The reason is that both AODV and DSDV prefer fresh routes to short ones.

Considering the delivery ratios under false distance vector attacks, we find that in both protocols they drop to around 50% to 60%. It is determined by the characteristic of distance vector mechanism. If the attacker can accurately predict the sequence number of node 29, the probability that a node will be cheated depends on the probability that it is closer to the attacker than to the victim. In this environment, it is 50% because the movement of every node is independent. Since the attacker applies a conservative method to predict the sequence number of the victim, the delivery ratio is a little higher than 50%.

The difference between the delivery ratios of ADOV and DSDV when they are under false destination sequence attacks is caused by the implementation of the attacker behaviors. In AODV, the malicious node will add a constant value to the sequence number carried in corresponding RREQ and uses the result as the sequence in false RREP. We choose the constant as 2 in the simulation. So there are chances that the false sequence

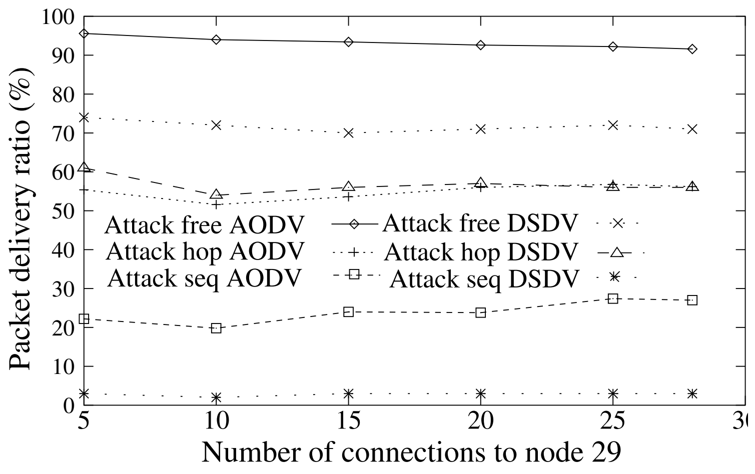


Figure 5.1. The delivery ratio versus the number of connections

cannot beat the real number. In DSDV, once the false sequence has been established, the false route propagates throughout the network. So more nodes will be cheated. If in AODV the attacker uses a very large number as the false sequence (e.g. 0x7fffffff), we would expect a lower delivery ratio.

One interesting point, when AODV is under attacks on destination sequence, is that the delivery ratio will increase a little as the number of connections increases. It happens because the attacker only adds a constant to the sequence in RREQ. As the number of connections increases, the true sequence increases faster, and the probability that the chosen fake sequence is smaller than the real value also increases. Thus less traffic will be attracted to the attacker.

Figure 5.2 shows the number of nodes that are cheated by the false routes when the number of connections increases from five to twenty-eight. In DSDV, the number of nodes that are cheated does not vary a lot as the number of connections changes because of the proactive property. When false distance vector attacks are conducted, less than half of the nodes are cheated. But when the network is under false destination sequence attacks, almost all nodes are cheated. In AODV, as the number of connections increases, more false RREP will be sent by the attacker. Therefore, more innocent nodes will be cheated. Both protocols prefer the route with larger sequence numbers, so the false destination sequence

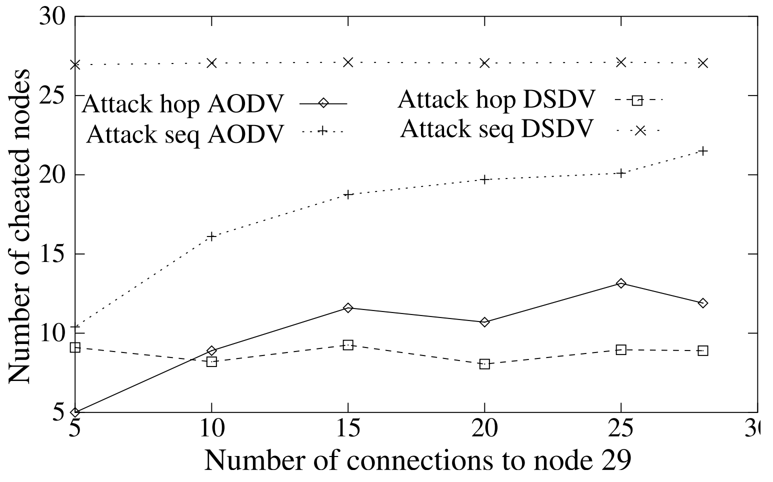


Figure 5.2. The number of cheated nodes versus the number of connections

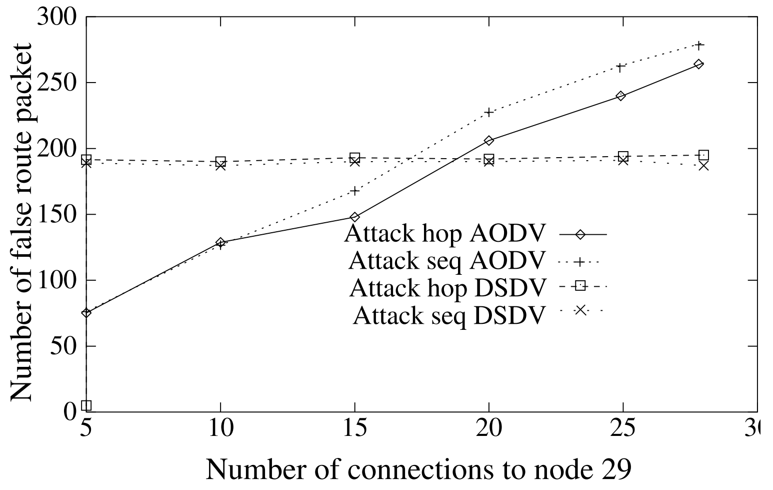


Figure 5.3. Number of false route packet versus the number of connections

attacks cheat more nodes. If the nodes are uniformly distributed in the test area, there are about half of the nodes that are closer to the attacker than to the destination if the nodes are uniformly distributed. They will be cheated by the false distance vector attacks if the sequence numbers in false routes are the same as the real ones. Since the attacker applies a conservative sequence prediction method, there are less than 50% of the nodes that are cheated in both protocols.

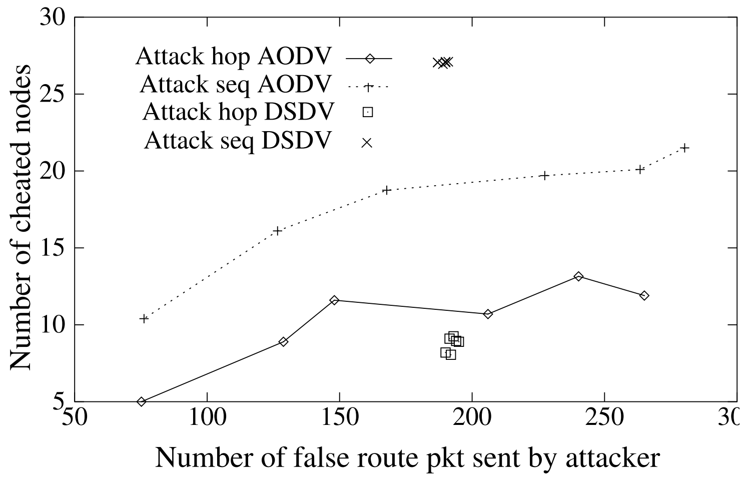


Figure 5.4. The number of cheated nodes versus the false route packet

Figure 5.3 shows the communication overhead of the two attacks. The number of false route updates sent in DSDV does not change a lot because of the proactive property. And the overhead of conducting two attacks does not show big difference. In AODV every false RREP can only attack one RREQ, so the number of false RREP sent by the attacker is roughly proportional to the number of connections. The two curves for AODV are very close to each other, which shows that both attacks put similar traffic overhead on the attacker. But the one for false destination sequence attacks is a little higher. It is because the false sequence numbers generated by the attacker disturb the updates to real numbers and introduce more route queries into the system.

Figure 5.4 examines the efficiency of the two attacks in both protocols. It shows the number of nodes that are cheated versus the number of false route packets sent by the attacker. For DSDV, the values form two groups of points which are very close to each other. They can be derived from the curves shown in Figure 5.2 and 5.3. For AODV, the curves are very similar to the lines in Figure 5.2 because the number of false RREP packets sent by the attacker is roughly proportional to the number of connections.

From Figure 5.1 to 5.4, we can tell that the attacks on destination sequence and the attacks on distance vector have similar communication overhead but the former ones have more severe impacts. For the intrusion prevention and intrusion detection systems de-

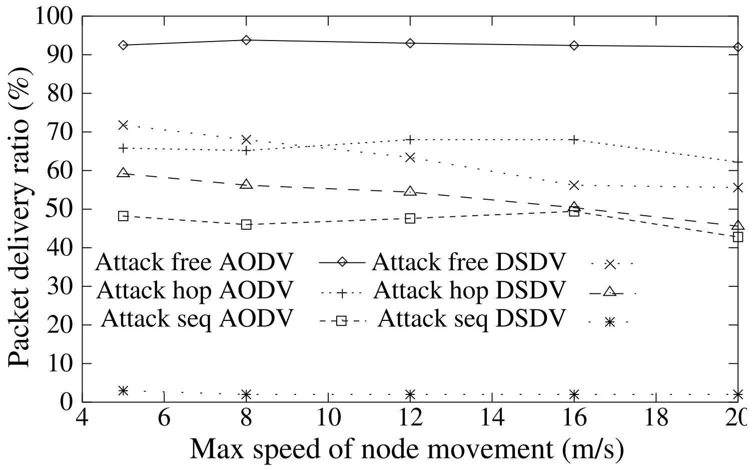


Figure 5.5. The delivery ratio versus the mobility of nodes

signed to protect ad hoc networks using AODV or DSDV, this kind of attack should be considered first.

5.4.2 Simulation condition two

Under condition two, we generate a scenario that contains twenty-nine connections. Each innocent node is the source of one connection and the destination of another. Node 5 sends false routes about all other nodes. We study the selected parameters versus the mobility of the nodes, which is represented by the maximum moving speed.

Figure 5.5 shows the delivery ratio versus the maximum speed of nodes under the conditions the same as Figure 5.1. The delivery ratio of attack free AODV keeps high, which shows that the mobility of node is still within the suitable serving range of AODV. DSDV has a slower response to link changes caused by node movement, so the delivery ratio decreases faster. When the malicious node conducts false destination sequence attacks on DSDV, the false routes will propagate throughout the network. Most of the innocent nodes will be cheated. So the delivery ratio will be low. Comparing to Figure 5.1, we find that more data packets successfully reach to the destinations when AODV is under attack. This can be explained by the difference between the connection scenarios of the two test cases.

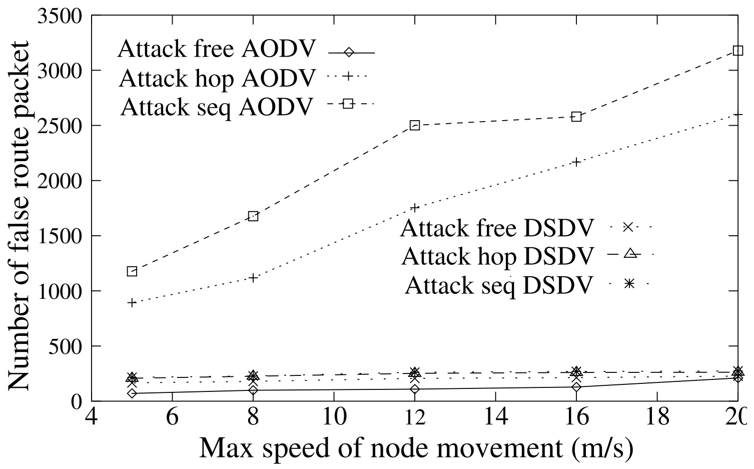


Figure 5.6. The attack overhead versus the mobility of nodes

Under condition two, every node is the source of one connection, and it may broadcast the RREQ throughout the network. Other nodes can establish the routes through the paths that they receive the request. Therefore, many nodes do not have to listen to the false RREP sent by the attacker. More safe routes are set up and the delivery ratio is higher.

Figure 5.6 shows the number of routing packets sent by node 5 when it behaves properly and when it conducts the attacks. In DSDV, the curves are very close to each other because the attacker can carry multiple false routes in the same routing packet. It does not have to increase the frequency of sending route updates. In AODV the attacker will send five to ten times more RREP when it attacks every RREQ it receives. It is not efficient for an aggressive malicious node to attack all connections at the same time in a reactive protocol. This anomalous increase may also be detected by IDS. Further research is required on the behaviors of intelligent attackers and suitable responses.

5.5 Detecting false destination sequence attacks

When the malicious nodes introduce false information into the networks, their behaviors and the conflicts between false and true information form special patterns, which can be used to detect the attacks. In addition, the connectivity history and the propagation

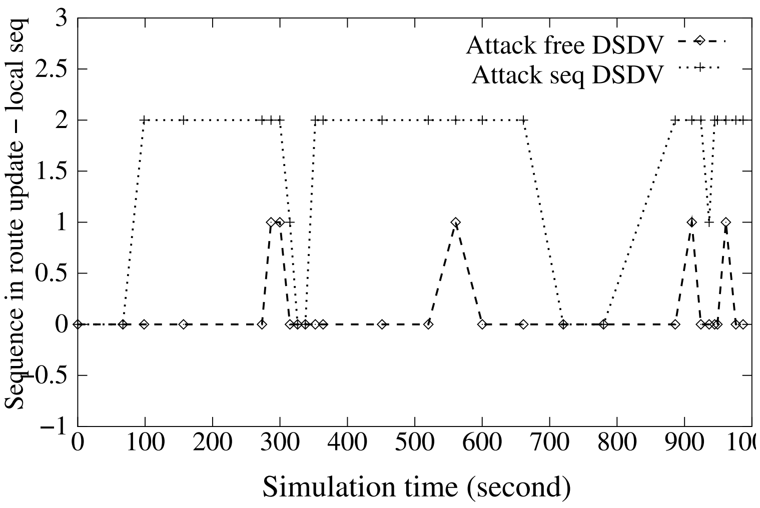


Figure 5.7. The anomalous patterns of sequence number in DSDV

paths of the false information can be used to identify the sources of attacks. Our research on security in ad hoc networks [124] tries to collect patterns of attacks and to provide guidelines for the design of the IDS. An example of detecting false destination sequence attacks in AODV and DSDV is given out below.

False destination sequence attacks can cheat a large part of the nodes and severely impact the delivery ratio. To "beat" other available routes, the attacker must choose a number, which is larger than the sequence generated by the real destination, as the false sequence to show its "freshness". If the victim can find this false sequence number, it will detect the attack. In DSDV the false sequence route will be transferred to all directions. There exists an upper limit of delay that the false route will reach to the victim if it is connected to the attacker and the false sequence always outruns the real one. In AODV, only when a node on the false route moves out of the range of its neighbor, the re-initiation procedure of the source will send out RREQ that carries the false sequence. Because the RREQ is broadcast throughout the network, there is a good chance that the real destination will receive the request. If the false sequence is still larger than the real one, the node detects the attack. Therefore, no upper limit of delay between the attack is conducted and it is detected can be guaranteed.

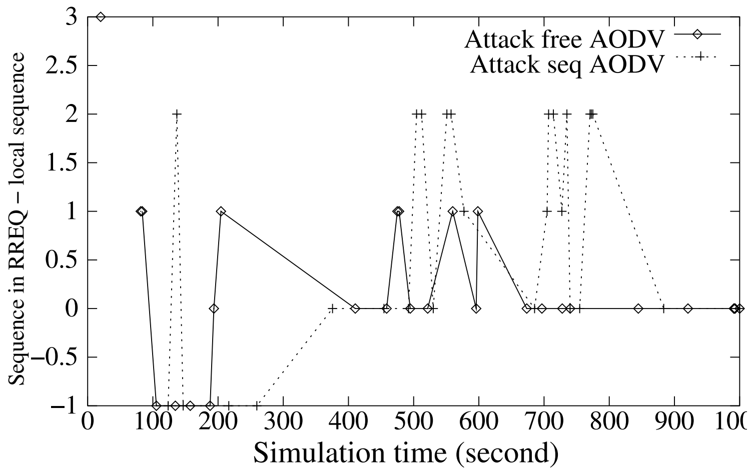


Figure 5.8. The anomalous patterns of sequence number in AODV

The sequence number of a node is carried in the routing packets. Under the normal operation of AODV and DSDV, the sequence carried in the packets can never be larger than the real sequence plus one. But when the node is under false destination sequence attacks, the difference between the received and local sequence numbers can be larger than or equal to 2. Figure 5.7 and 5.8 show the difference between the two sequence numbers detected by a node when all other nodes behave properly and when one malicious node attacks it with false destination sequence. In DSDV, when the false sequence is larger than the real number, it can be detected in any route update sent by a neighbor of the victim. If the real number is larger, the attacker will find it and conducts the new attack. Therefore the difference fluctuates between 0 and 2. In AODV, the innocent node detects eleven times that the incoming sequence number is larger than local number plus one.

In both scenarios some attacks are not detected. Two problems that impact the detection of false destination sequence attacks on AODV and DSDV are: (1) The real sequence may outrun the false one when it is received by the victim. (2) A tight limit of the delay between the false sequence is generated and it reaches the victim, if the two nodes are connected, should be achieved. A protocol that uses one detected attack to activate the detection of other attacks has been designed [124]. The basic idea is to re-examine all routing information coming from the same sources and activate the re-initiation. A software module that can be integrated into AODV and DSDV to improve the detection accuracy is under construction.

Collecting and determining the anomalous patterns of attacks is a challenging topic in IDS for ad hoc networks. The example provided above shows that combining protocol analysis and practical simulation may accelerate this procedure.

5.6 Conclusion

The security of ad hoc routing protocols is still an open problem and deserves more research efforts. This chapter studies the vulnerabilities of and attacks on two protocols – AODV and DSDV. The analysis shows that as AODV provides fair performance with reasonable overhead and adaptability to both traffic load and node mobility, the on-demand property introduces some security deficiencies. It allows the malicious node to attack the network in real time with flexibility. It is more difficult to locate the sources of the false information. The proactive property also has disadvantages. The routine exchange of routes enables the false routing information to propagate within a wider range. The malicious node can conduct multiple attacks in the same routing packet. Since both protocols prefer the fresh routes which are identified by large sequence numbers, the attacks on destination sequence have more severe impacts than the attacks on distance vector.

The simulation supports our analysis. The delivery ratio curves show that the attacks on destination sequence will attract more packets to the attackers. False distance vector attacks will cheat less than 50% of the nodes in a uniformly distributed network. The communication overhead caused by conducting attacks is more stable on the traffic load and the width of attacks in DSDV than in AODV. The analysis and simulation also show that it is more efficient to detect false destination sequence attacks in DSDV than in AODV.

The research to protect wired network routing protocols [31] has shown that it is the property, instead of the protocol detail, that leads to the security deficiencies. The example attacks on AODV and DSDV can also be applied to attack other protocols sharing the properties. Thus the analysis results and anomalous patterns of the attacks can be employed to prevent or detect the coterminous attacks on different protocols. Because the primary difference between AODV and DSDV is the on-demand and proactive properties, we may generalize the analysis to other on-demand or proactive protocols.

There are many problems to be solved in protecting ad hoc networks. We plan to study the relationship between the average delay of detecting false destination sequence attacks and the mobility of the nodes. We plan to study more features of the routing protocols to exploit their security deficiencies. On achieving the secure distribution of individual intrusion detection result, we plan to establish the trust relation among nodes in the open area of ad hoc networks [126]. The results will provide guidelines for the design of a secure routing protocol and become the building blocks of the IDS for ad hoc networks.

6 DEFENDING AGAINST FALSE DESTINATION SEQUENCE ATTACKS IN AODV

6.1 Introduction

A mobile ad hoc network is a collection of wireless nodes that can be rapidly deployed as a multi-hop packet radio network without the aid of any established infrastructure or centralized administration [127]. Such networks can be used to enable next generation of battlefield applications envisioned by the military, and provide communication for civilian environments. Since the routing protocols for ad hoc networks are vulnerable to malicious attacks, security and protection mechanisms must be adopted to guarantee the fair performance in the applications.

Intrusion detection mechanisms have been introduced into the protection of ad hoc networks as the second line of defense in addition to the prevention methods. The difficulties to apply Intrusion Detection Systems (IDS) to ad hoc networks were discussed by Zhang and Lee [15]. Bhargavan [16] and Haas [17] explored the security issues in wireless LANs and ad hoc networks respectively. Several security enhancements and IDS structures have been presented by Bharghavan [122], Haas [33], and Albers [18].

However, just detecting attacks, without identifying and isolating the attackers, leaves the protection to ad hoc networks in a passive mode. First of all, it cannot protect the innocent nodes from getting cheated by the same adversary. The malicious node can either choose a new kind of attack or roam to another part of the network and conduct the same attack on its new victims. The IDS agent on every node has to monitor traffic without emphases.

Second, the IDS could have benefited from the discovered attacks to accelerate the detection of other underway attacks. An adversary usually conducts multiple attacks at the same time. If the source of an attack is located, the innocent nodes can detect more attacks from the same source by collecting and examining the information provided by it. Both efficiency and accuracy of the IDS will be improved.

For example, most ad hoc routing protocols are cooperative in nature [128]. An adversary who hijacks a mobile node can disseminate false routing information about multiple victims and paralyze the network. If one of the false routes is detected without exposing the source, unless every node in the network re-establishes the connections and gets certified response from the destination, other false routes cannot be excluded. But we know that the multiple false routing trees are all rooted at the malicious node. If the attacker is identified, all routing information from it will be re-examined and it will be much easier to discover other false routes.

Therefore, an active intruder identification procedure will help all innocent nodes to pinpoint the malicious nodes, thus preventing them from introducing new attacks. The extra costs on communication and computation caused by the identification procedures pay back on providing a more secure network and better performance.

In this chapter, we present a protocol, Reverse Labeling Restriction (RLR), to identify and isolate the malicious nodes that conduct false destination sequence attacks on AODV. RLR identifies the upstream of the propagation path of the false routes through reverse labeling. The victims broadcast the suspicious attackers and activate the route reinitiation to accelerate the intruder identification procedures. The information from various nodes is integrated through quorum voting. We show by simulation that when RLR is integrated with AODV, most innocent nodes can successfully identify all malicious nodes, and the isolation of attackers leads to significant performance improvements with regard to packet delivery ratio. The robustness of the proposed mechanism is analyzed.

The remainder of this chapter is organized as follows: Section 6.2 presents the background of intruder identification and sets up the criteria to evaluate the algorithms. Related work on intruder identification in wired networks is discussed. Section 6.3 presents the details of the proposed mechanism. Section 6.4 illustrates the simulation results on detection accuracy and communication overhead. Section 6.5 presents the robustness analysis of the proposed mechanism. Section 6.6 concludes the chapter.

6.2 Background

We begin with the objective of intruder identification. The criteria to evaluate the algorithms in ad hoc networks are presented. Although a lot of research on intruder identification in wired networks has been accomplished, the difference between the two kinds of network environments determines that few mechanisms can be ported without efforts.

6.2.1 Evaluation criteria

Intruder identification in ad hoc networks describes the procedure of identifying the users or nodes that conduct anomalous activities that threaten the connectivity and reliability of the networks and the authenticity of the information. We have established four criteria for the evaluation of the intruder identification algorithms: accuracy, overhead, effectiveness, and robustness.

- Accuracy: Accuracy can be defined in terms of two notions derived from intrusion detection. False positive mistakes describe the number of innocent nodes that are incorrectly marked as malicious. False negative mistakes describe the number of malicious nodes that are not identified as such. If an algorithm is too lenient, false negative ratio is high, and the un-identified attackers will form new attacks on the network. On the contrary, if a strict algorithm leads to high false positive ratio, many innocent nodes will become suspected. Under the extreme conditions, no routes that do not pass any suspected nodes can be discovered and the network is paralyzed.

- Overhead: The overhead describes the increases in control packets and computation costs for identifying the attackers. Mobile nodes usually have weak processing capability, limited storage space, and narrow bandwidth. If the identification algorithm is not designed properly, it will exhaust all available resources and leave the network secure but useless. Since we assume that intruder identification will be activated only when an attack is detected, it will not introduce much overhead when the network is operating normally.

- Effectiveness: Effectiveness describes the improvements in the ad hoc network performance after the malicious nodes are identified and isolated. The example metrics include the increase of the packet delivery ratio, the decrease of average delay, and the decrease of normalized protocol overhead. Effectiveness is an independent criterion from accuracy or overhead. For example, sharp increase in performance does not necessarily imply low false coverage. The ultimate objective of intruder identification is to enable secure network topology, thus achieving fair performance. Effectiveness is established to evaluate the algorithm's capability to satisfy this requirement.

- Robustness: The intruder identification algorithm can become the target of attacks. For example, sharing fake communication traces and impersonating identification results can lead to network partition and result in performance degradation. The robustness evaluates the algorithm's capability to avoid being compromised.

The design of an algorithm based on the above criteria will lead to an efficient and low-cost intruder identification protocol. When the characteristics of the applications are considered, some tradeoffs, such as reducing the false coverage ratio and lowering the identification overhead, must be reached through careful design.

6.2.2 Previous research on intruder identification

Locating the sources of the attacks and isolating the malicious nodes from the rest of the network has been proven to be an efficient mechanism to protect the connectivity and the applications. The severe impacts of the attacks on network performance have inspired the research on intruder identification and brought several solutions in wired networks.

Non-repudiation was one solution to prevent the malicious nodes from denying that they have distributed certain information [129, 130]. Unforgeable digital signature is sent along with the information. When later the intrusion is detected, the recipient can use the signed information to identify the attacker and prove it to all other nodes. The solution is neat but with heavy computation overhead, especially for the routing packets that have to be signed by every node along the path. It is difficult to apply this method to ad hoc networks when the limited computation resources available to the mobile nodes are considered.

Probabilistic packet marking and source traceback has been introduced especially to identify the sources of the distributed Deny-of-Service (DDoS) attacks [131, 132]. Each router probabilistically inscribes its local path information onto a traversing packet so that the destination node can reconstruct the complete path by examining a large number of packets. There are some difficulties to apply this method to ad hoc networks. The marking procedure will increase the processing overhead at every node. And this mechanism cannot locate the malicious node if it does not conduct DDoS attack. Most importantly, the research by Park $et\ al$ [133] has shown that this method is vulnerable to the injected false tracing information.

Active edge-tagging is another approach to identify and isolate the sources of DoS attacks [134]. It is based on the active network technology, which provides a dynamic execution environment for the monitoring programs and enables fast deployment of new protocols without modifying network infrastructure. Enabling the remote process to execute on local node will introduce new security challenges to ad hoc networks.

Although a lot of research on intruder identification in wired networks has been conducted, few mechanisms can be directly ported to ad hoc networks. The research challenges in ad hoc networks must be examined and the new protocols to solve this problem must be designed.

6.3 Reverse labeling restriction protocol

6.3.1 System assumption

We assume that every mobile node in the network can be uniquely identified by its ID. The ID of a node is combined with a public-private key pair which can be used for its digital signature. The ID of a node is used during the intruder identification procedures. Since the asymmetric encryption is very computational expensive for the mobile devices, we also assume the establishment of pair wise keys among the wireless nodes. The previous approaches such as [70, 135] can be used to accomplish the task.

In the proposed mechanism, we assume that every node uses a local blacklist to record the nodes that it suspects. The node has a total control on adding and deleting entities from its list. For the clarity of the remainder of this chapter, we call the real attacker as a "malicious node", while the nodes in blacklists are called "suspected nodes".

We assume that a malicious node has a total control on the moment of time, the target, and the mechanism of an attack.

The above assumptions describe a well-defined ad hoc network environment for intruder identification. It is also general enough to include most of the typical attack scenarios. No assumption stated above excludes any specific protocols. Therefore, the algorithms designed under these assumptions can be easily applied to different ad hoc environments.

Since ad hoc networks do not have centralized administration, every node can make its own decision on which entitles are suspect. However, the intruder identification procedures will conclude when most, if not all, innocent nodes in the network reach an agreement on the identity of the attacker. Mechanisms must be designed to accelerate the identification procedure.

6.3.2 Basic Ideas

The routes to a specific node in an ad hoc network using AODV will form a routing tree rooted at the destination. When the node is under false destination sequence attacks, the malicious nodes will break the routing tree into a forest, as shown in Figure 6.1, in which the attackers will be the roots of the false routing trees. The procedure to locate these roots is the procedure to identify the malicious nodes.

Since we assume that the pair wise keys have been established among the wireless nodes, a mobile node can verify the identities of its direct neighbors. It is safe for a node to conclude that the entity that provides the false routing information is one step closer to

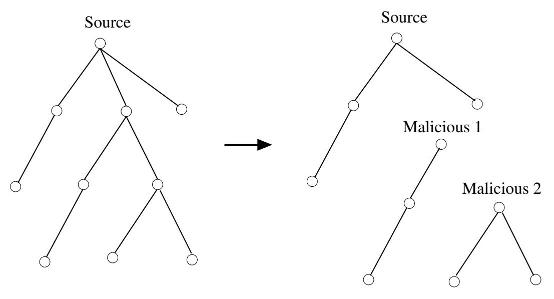


Figure 6.1. Routing forest of a mobile node with false routing trees

the attacker. Therefore, the nodes that are physically closer to an attacker have a higher probability to show up in multiple false routing trees than the nodes that are faraway. If the mobile nodes share their information about the previous hops along the false routes, either the attacker or the nodes that are very close to it will appear most frequently. This enables the mobile nodes to use both local knowledge and global collaboration to identify the attackers.

We assume that every node maintains a blacklist to record the entities that it suspects. In RLR, when a node discovers anomalous destination sequence numbers from the incoming RREQ as described in Section 5.5, it detects the false destination sequence attacks. It will broadcast a packet with the content {node ID, current sequence, new sequence, local blacklist, digital signature}, which is called the "INVALID" packet. The objectives of this packet are two folds: (1) refresh the destination sequence number of the node and the routes to it, (2) activate the intruder identification procedures in other nodes.

Every node that receives this packet will examine its routing entry to the sender. If a sequence number larger than the "current sequence" in the packet is discovered, this is a false route and the node will put the previous hop along this path into its local blacklist. At the same time, every node will examine the entities in the blacklist in the broadcast packet and update their voting counters. If more than a quorum of nodes have declared that a specific node is suspicious, it will be put into local blacklist. The routing information from the suspected nodes will be rejected to achieve isolation. One important feature of RLR is that every node updates its blacklist independently.

RLR experiences false positive mistakes. If the nodes cannot be removed from a local blacklist, sooner or later every node will become suspicious and a totally partitioned network will be resulted in. To prevent such conditions from happening, we attach a timer to every node in the blacklist. When the timer expires, the node will be released. The duration of the timer will increase exponentially to the number of times that the node has been put into blacklist. More details of RLR will be discussed in the following section.

6.3.3 Protocol details

Updating blacklist by INVALID packet

A node can be put into a local blacklist under one of the two conditions: (1) it provides false routing information, (2) more than a threshold number of not suspected nodes have declared that this node is suspicious. In this section, we discuss the first condition. The second scenario will be studied later.

When a node receives an "INVALID" packet and finds that it has a false sequence number in the routing table, it will label the provider of the information as suspicious and put it into the blacklist. This labeling procedure will be conducted in the reverse direction of the propagation of the false route. Since the previous hop could only be an intermediate node instead of the attacker, some methods must be adopted to reduce the false positive mistakes in the protocol. In RLR, whenever a node is put into the blacklist because of providing false routing information, a timer will be attached to it. When the timer expires, the node will be released from the blacklist. We set the initial expiration duration to 60 seconds in our simulation because it is roughly the time that a node moves out of the range of its neighbors. Every time a node enters the blacklist, the timer coupled with it will be doubled. Therefore, if a node repeatedly provides false information, it will stay in the blacklist longer and longer. Through rejecting the routing information from the suspicious nodes, we restrict the false information from the attacker within a limited area.

The labeling procedure and the routing information rejection lead to the name "Reverse Labeling Restriction".

Every INVALID packet is uniquely identified by a couple $\{$source node ID, broadcast sequence$\}$. In our implementation, it shares the broadcast sequence with RREQ packets. The digital signature prevents an attacker from sending false INVALID packets in other node's name. A list is maintained by every node to guarantee that a given INVALID packet is only processed and rebroadcast once. In the INVALID packet, the sender will choose a new destination sequence number that is larger than any known false sequence numbers, so that the new routes can be established. Under the scenario in which the false sequence has already reached the largest number, the node will wait for a DELETE_PERIOD, defined by AODV, before sending INVALID to make sure that all inactive routes have been cleaned out from routing tables.

Updating blacklist by quorum voting

In the previous section, when a false sequence number is detected, only the nodes in the false routing trees will participate into the intruder identification procedure and benefit from it. We know that by sharing the local identification results, we can accelerate the procedure of reaching the consistent conclusion on intruders when the integrity and authenticity of the information is properly protected. To support such acceleration in RLR, a node will attach its blacklist to the INVALID packet so that a quorum voting can be conducted. The votes from a suspicious node will be ignored.

When a node receives the INVALID packet, it will examine the enclosed blacklist and update the voting counter for every node in it. If more than the threshold number of different nodes have declared a specific node to be suspect, that node will be permanently put into the local blacklist and the routing information from it will be rejected. The choice of the quorum value has a direct impact on the detection accuracy and the system performance. We refer to [9, 136] for the choice of this value and the tradeoff between the reaction to the attackers and the protection of legitimate nodes from false accusation.

Route rediscovery

When a node is added to the blacklist, the local host will traverse its current routing table and abort all entries using this node as the next hop. This operation is an attempt to disable all routes passing through the suspected nodes. RERR will be sent to previous hops along these paths and the reinitiation procedure will be activated. If there are other false routing trees rooted at the same malicious node, there is a high probability that they are discovered by this procedure.

Updating routes by INVALID packet

Since the INVALID packet is signed, all other nodes should update their routes to the sender to the paths through which they received the INVALID packet unless the previous hop is in their blacklists. Since the INVALID packet is broadcast throughout the network, every node can establish the new route unless it is partitioned or surrounded by malicious attackers.

Introducing the INVALID packets also provides a solution to the problem of aborting the maximum sequence number. When the destination sequence in AODV reaches its maximum value and rounds back to zero, the new sequence number will be rejected because it is less than the old value. Using a single INVALID packet, we can establish the new sequence and abort the old ones. The packet uses the maximum value as the old sequence, and zero as the new one. Since there is not any node that can hold a sequence larger than the maximum value, nobody will be put into the blacklists. At the same time, the new sequence will be established.

Packets from suspicious nodes

In this section we discuss the processing of packets when they come from a node in the blacklist.

- All data packets, HELLO packets, and RERR packets will be processed as usual. Data packets do not provide routing information. HELLO packets are used to maintain direct neighbor relationship. The abortion and reinitiation of routes caused by RERR may expose other false routing trees.

- If the packet is a RREQ and the source is not the suspect node, the request will be ignored. In this way we avoid building a reverse path back to the source through the suspect node. Because the RREQ is broadcast throughout the network, the local node has a good chance to receive the request through another exclusive path.

- If the packet is a RREP and the query destination is not the suspect node, the local host will ignore the reply. We do not trust any routing information from the nodes in the blacklist.

- When the packet is an INVALID, the remote blacklist is ignored if the sender is in the local blacklist. We examine the routing table and update blacklist if we detect any false sequences. No routes through the suspicious node should be established.

6.4 Simulation results

6.4.1 Simulation environment

The simulation of RLR in AODV is deployed using ns2 with CMU extension blocks for ad hoc networks. Two new modules, one for intrusion detection and the other for RLR, are implemented. Table 6.1 lists the simulation parameters that we use.

We examine the performance of RLR in different environments by varying node mobility and the number of independent attackers. We choose to use fixed maximum speed and describe the node mobility with different pause time. The number of independent attackers is chosen to evaluate the scalability of RLR. We do not use a high packet rate so that the packet loss caused by congestion can be avoided.

The following parameters are selected to evaluate the proposed mechanism: (1) packet delivery ratio, (2) the number of innocent nodes that identify all malicious attackers, (3) the number of innocent nodes that are incorrectly labeled as attackers, (4) the normalized

Table 6.1
Simulation parameters

Simulator	ns-2
Simulation duration	1000 seconds
Simulation area	1000 * 1000 m
Number of mobile nodes	30
Transmission range	250 m
Movement model	Random waypoint
Maximum speed	5 m / s
Number of CBR connections	25 / 50
Data payload	512 bytes
Packet rate	2 pkt / s
Node pause time	0, 10, 20, 30, 40, 50, 60 seconds
Number of malicious nodes	1 – 5
Expiration period	60 second
Quorum voting threshold	3

protocol overhead, (5) the average number of signed packets to be processed by every node.

We use metric (1) to evaluate the effectiveness of RLR. The simulation has shown that one malicious node attacking the network with false destination sequences may cause 50% of the data packets to be dropped. We expect the adoption of RLR to bring a sharp increase in the packet delivery ratio. Metric (2) and (3) evaluate the detection accuracy of RLR. They examine the false negative and false positive ratio respectively. Metric (4) and (5) evaluate the overhead of the proposed mechanism. They examine the communication and computation overhead respectively. The normalized protocol overhead is calculated as the number of control packets on every delivered data packet. Every data point shown in the following figures is the average value of ten runs of different node movement scenarios.

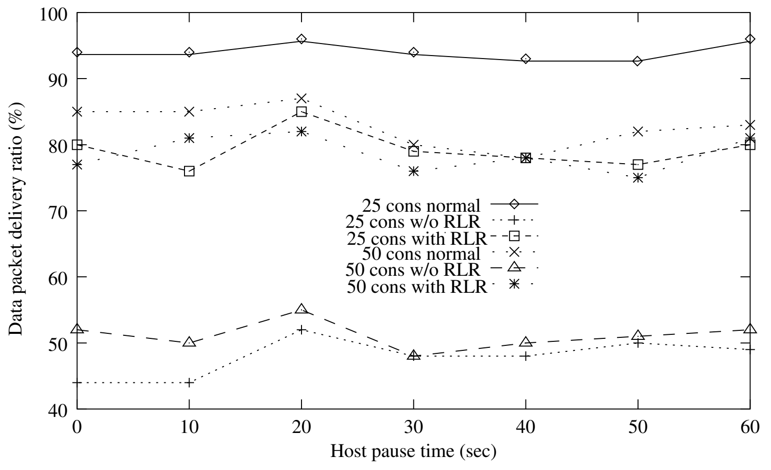


Figure 6.2. Data packet delivery ratio versus node pause time

6.4.2 Effectiveness

Figure 6.2 illustrates the packet delivery ratio versus node pause time when there is only one malicious node in the network. The three conditions are: (1) all nodes are innocent, (2) one malicious node exists and RLR is disabled, (3) one malicious node exists and RLR is enabled. Two traffic load scenarios with 25 connections and 50 connections among the 30 nodes are studied. The packet delivery ratio gets an average increase of about 30% in both traffic loads when the network is protected by RLR. It does not vary a lot with the changes of pause time because the high node mobility puts challenges to the attacker as well. When the pause time is short, the frequency of route changes caused by node movement is high. More route re-initiations will be activated and more false routing trees will be detected. Therefore, the procedure of identifying the malicious node converges quickly. When the pause time increases, the correct routes established by the INVALID packets become stable. Under both conditions, we may expect a significant improvement in packet delivery ratio when RLR is applied.

Figure 6.3 illustrates the packet delivery ratio versus the number of malicious nodes. The node pause time is 30 seconds. Since the malicious nodes will compete with each other, only the routes with the largest false sequence numbers will be adopted. Before

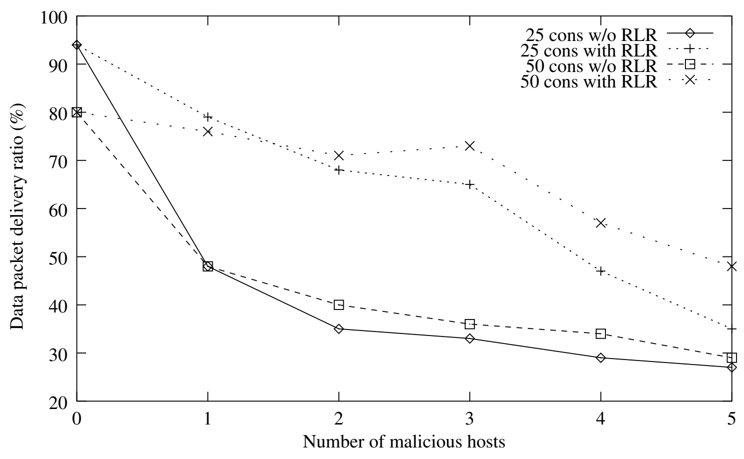


Figure 6.3. Data packet delivery ratio versus the number of independent attackers

all of the malicious nodes are identified and isolated, the data traffic will be attracted to the attackers and get discarded. This explains the decrease in the delivery ratio within the simulation duration when there are multiple malicious nodes. When there are more connections, more false routing trees will be discovered. The identification procedure converges in a shorter time, thus resulting in a higher delivery ratio.

6.4.3 Detection accuracy

Table 6.2 and 6.3 show the detection accuracy of RLR versus node pause time when there is only one malicious attacker in the network. RLR has a low false negative ratio when the pause time changes, which shows that the proposed protocol has a good detection accuracy when the moving speed of nodes is not high. When there are more connections in the system, more false routing trees will be detected. This explains the reason that within the simulation duration more innocent nodes successfully identify the malicious attacker when there are 50 connections in the network. However, the false positive ratio has a sharp increase when the number of connections changes from 25 to 50 because more INVALID packets and remote blacklists are broadcast. If we use the same threshold for quorum voting, more innocent nodes will be put into blacklists by mistake. The tradeoff between false positive and false negative ratios will be studied in future research.

Table 6.2
Detection accuracy of RLR versus node pause time

	29 innocent nodes, 25 connections	
pause time (sec)	# of innocent nodes identify the attacker	average # of innocent nodes marked as attackers
0	24	0.22
10	25	0
20	24	0
30	28	0
40	24	0
50	24	0.07
60	24	0.07

Table 6.3
Detection accuracy of RLR versus node pause time

	29 innocent nodes, 50 connections	
pause time (sec)	# of innocent nodes identify the attacker	average # of innocent nodes marked as attackers
0	29	2.2
10	29	1.4
20	25	1.1
30	29	1.1
40	29	0.6
50	29	1.1
60	24	1.0

Table 6.4 and 6.5 illustrate the detection accuracy of RLR versus the number of malicious nodes. As the number of attackers increases, fewer innocent nodes are able to identify all of them during the simulation duration and the packet delivery ratio is impacted. Lowering the threshold values may accelerate the detection procedure, but it will lead to a high false positive ratio.

Table 6.4
Detection accuracy of RLR versus number of malicious nodes

	25 connections	
number of malicious nodes	# of innocent nodes identify all attackers	average # of innocent nodes marked as attackers
1	28	0
2	28	0.65
3	25	1
4	21	0.62
5	15	0.67

Table 6.5
Detection accuracy of RLR versus number of malicious nodes

	50 connections	
number of malicious nodes	# of innocent nodes identify all attackers	average # of innocent nodes marked as attackers
1	29	1.1
2	28	2.6
3	27	1.4
4	25	2.2
5	19	4.0

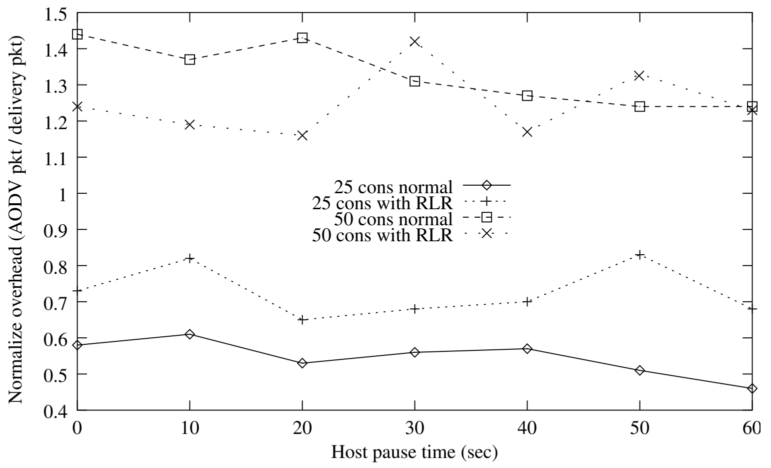


Figure 6.4. Normalized control packet overhead versus node pause time

6.4.4 Overhead

Figure 6.4 illustrates the normalized control packet overhead versus the node pause time when the network is under attack but protected by RLR. As the baseline for comparison, we also show the data when no attack exists and the RLR is disabled. We find that under two conditions the overhead does not show big differences. Combined with Figure 6.2, we note that RLR can improve the performance of ad hoc networks with a limited cost when they are under false destination sequence attacks. One interesting point is that when there are 50 connections, the normalized overhead with RLR for most data points is lower than in normal conditions. The reason is as follows. When the number of connections increases, there are more overlapped source and destination nodes. Therefore, when an INVALID packet is sent out, multiple connections will benefit from the fresh routes. It reduces the routing queries, and results in lower normalized overhead.

Figures 6.5 and 6.6 illustrate the computation overhead (caused by signing and verifying signatures) of RLR versus the node pause time and the number of malicious attackers. The computation overhead does not vary a lot when node pause time changes. When the number of attackers increases from 1 to 5, the computation overhead increases slowly, which shows good scalability.

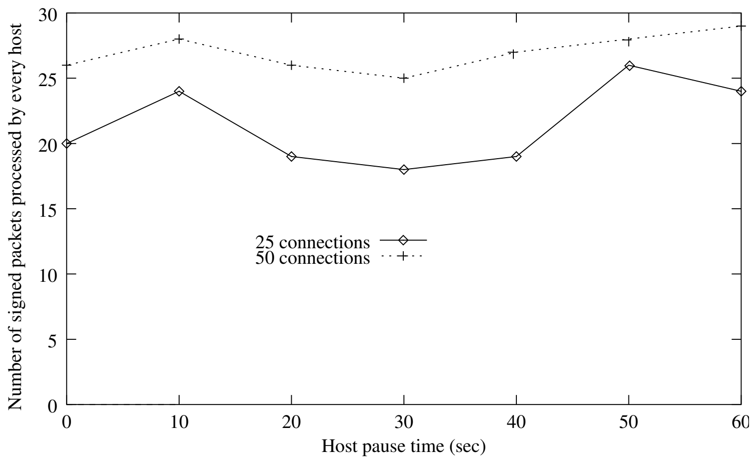


Figure 6.5. Computation overhead of RLR versus node pause time

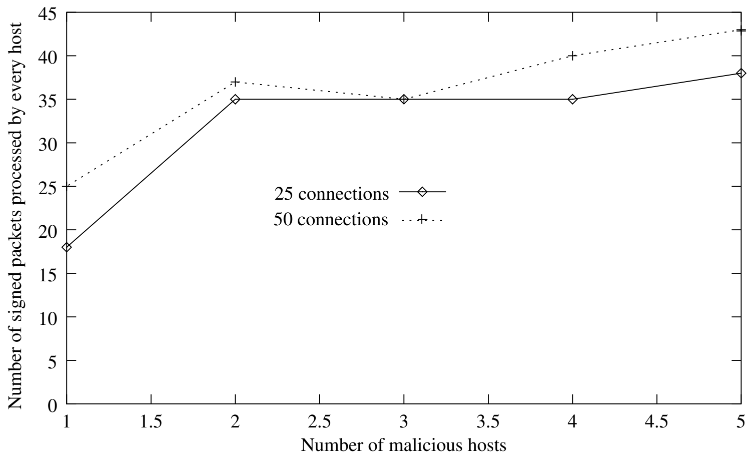


Figure 6.6. Computation overhead of RLR versus the number of attackers

6.5 Discussion

In this section, we examine the robustness of the proposed mechanism under several attacks.

False INVALID packets

As we have discussed in the previous section, the INVALID packet is protected by the digital signature of the source. This prevents an attacker from sending INVALID packets in other node's name. It also prevents the malicious nodes from using partial information to conduct the re-transmitting attacks.

The malicious nodes may discard the INVALID packets to avoid being identified. RLR does not suffer from this kind of passive attack because the INVALID packets are broadcast throughout the network. As long as there exists a path from the source not passing through any malicious nodes, the local host will receive the INVALID packet.

A malicious node can conduct active attacks by sending false INVALID packets in its own name. However, if all other nodes behave properly, the destination sequence of the malicious node can only be generated by itself. The reverse labeling procedure will converge at the attacker. If its neighbors are relatively stable, they will soon permanently place it in their blacklists, and the false routing information will be blocked. If the neighbors are constantly changing, the INVALID packets will expose the attacker to more nodes, and the quorum voting procedure will identify it later.

False blacklist

Another attack that the malicious node can conduct is to attach a false blacklist to an INVALID packet. As we discussed in Section 6.3.3, a quorum value is adopted to decide whether the nodes in remote blacklists should be permanently marked as suspicious. If the attacker has already been identified, its votes will be ignored. If it is not in the local blacklist, it can only reduce the quorum value by one. If the threshold is chosen as k, RLR is robust against the collusion of up to $k - 1$ compromised nodes.

Attack on a single node

The simulation results presented in Section 6.4 assume that the malicious nodes attack every RREQ they receive. Although the ad hoc network assumes a flat infrastructure, in real applications some special nodes may be more important than others. If the malicious nodes only attack the routes to the special nodes, it is more difficult to identify them.

When RLR is applied to such environments, the attacked nodes will detect the false routing information and broadcast the INVALID packets. As we have discussed in Section 6.3.3, most nodes will establish new routes to the attacked nodes through the INVALID packets. Before the paths are broken, the data packets will successfully reach their destinations.

6.6 Conclusion

RLR is a practical intruder identification protocol that protects AODV against false destination sequence attacks. It uses the reverse labeling procedure to trace back to the sources of false routing information. The RLR protocol combines knowledge from different nodes and achieves robustness against collusive attackers. RLR uses timer expiration to reduce the false positive mistakes.

The simulation results show that most of the innocent nodes equipped with RLR can identify all attackers. An average increase of 30% in packet delivery ratio is expected when the nodes do not move at a high speed. RLR exhibits good scalability with regard to the number of attackers. It brings all these benefits with a limited communication and computation overhead.

RLR implements an efficient and accurate intruder identification without losing generality. It is easy to integrate it with other routing protocols of ad hoc networks. For example, when malicious nodes attack the DSDV protocol with false distance vector or false destination sequence, the reverse labeling procedures can be used to trace back to the attackers. The propagation of the INVALID packets and the establishment of new routes can be accomplished by the routine routing information exchanges.

We plan to design more criteria to evaluate the intruder identification algorithms. They will provide a uniform standard to compare different protocols. We plan to study more features of the ad hoc networks to exploit their security vulnerabilities. The mechanisms that can efficiently extract related traces from data collector when an attack is detected will be designed. We also plan to investigate the possibility of applying fuzzy control methods [137] to local identification engine to reduce the false positive and false negative mistakes when the information is incomplete for a specific attack.

7 CONCLUSIONS AND FUTURE WORK

7.1 Conclusions

7.1.1 Wormhole detection in mobile ad hoc networks

We first establish a basis to identify the wormhole detection capabilities of the proposed mechanisms, so that our work can be distinguished from previous efforts. We classify the wormholes into three types: closed, half open, and open, according to whether the malicious nodes are visible on the route. The analysis shows that previous research focuses on the prevention of closed wormholes. The half open and open wormholes are more difficult to detect because:

- The mechanisms that only examine direct neighbors cannot guarantee the detection.
- The mechanisms that only examine a part of the intermediate nodes along the path cannot guarantee the detection.

Therefore, an end-to-end mechanism must be designed.

In the proposed end-to-end mechanism, only trust between the source and the destination of a route is assumed. The source and the intermediate nodes will attach their timestamps and positions to the wormhole detection packets so that the conflicting information sent by the attackers can be detected. The integrity and authenticity of the attached information is protected by the message authentication code. The end-to-end mechanism implements both individual packet and cross packet examination. One disadvantage is that it requires $O(l\ m)$ storage space and $O(l\ m^2)$ computation operations, where l is the length of the path, and m is the number of detection packets.

To prevent the destinations from being overwhelmed by the overhead of wormhole detection, an efficient information management scheme COTA is designed. To avoid recording and comparing all $\langle$time, position$\rangle$ pairs, the following methods are adopted:

- Divide the area into same-sized cells and time into same-length slots. Only the first ⟨time, position⟩ pair of a node that falls into the same cell and the same slot is recorded.

- The lifetime of a detection packet T_{life} is defined. The packets that have traveled in the network for a time longer than T_{life} will be discarded.

These methods have brought the following advantages: (a) They restrict the number of time slots that COTA needs to store for every intermediate node. (b) They restrict the longest moving distance of a node during the delivery of a detection packet. (c) For cross packet examination, the destination only needs to select from each cell a record of that node that has the shortest time difference from the new record. Therefore, the design goal to control the storage and computation overhead has been achieved.

To evaluate its detection capability, the $sensitivity$ of COTA is introduced into the mechanism. It is determined by the size of a cell, the maximum moving speed of the node, and the time slot length. Through adjusting the parameters listed above, the destination node can control the resource that it wants to consume on wormhole detection and the detection capability of COTA.

7.1.2 Wormhole detection in sensor networks

Previous approaches for wormhole detection in ad hoc networks usually depend on some special hardware such as GPS, synchronized clocks, or directional antennas, which will impact their adoption in sensor networks. We have proposed a family of approaches that integrate the techniques from social science, computer graphics, and scientific visualization to detect such attacks in sensor networks. They do not require the sensors to be equipped with any special hardware.

In MDS-VOW, the sensors first estimate the distances to the neighbors based on the received signal strength. The data then are transferred to a centralized controller who will reconstruct the network using multi-dimensional scaling. The analysis and experiments show that the wormhole bends the reconstructed network to pull the sensors to each other and fit the fake connections. This forms the two-ended torch structure that can be used to detect the fake neighbor connections. MDS-VOW consists of multiple steps and each step can be improved independently.

Experiments using grid placement and random placement of sensors are conducted to examine the detection accuracy of the proposed mechanism. The results show that when the distance estimation errors are uniformly distributed and the error rate is equal to or smaller than 0.6, MDS-VOW can detect most of the fake connections without introducing many false positive alarms.

As a distributed approach, Dis-VoW allows every sensor to conduct local network reconstruction and wormhole detection. The distortions in edge length and angles among neighboring sensors are used to calculate the normalized variable $wormhole\ indicator$ for every sensor which will be used to identify the fake neighbor connections.

Experimental studies using grid placement and random placement of sensors are conducted to examine the detection accuracy of the proposed mechanism. The results show that Dis-VoW can detect most of the fake neighbor connections without introducing many false positive alarms even when there are multiple wormholes in the network.

7.1.3 Security study of distance vector routing protocols

The research compares the security properties of AODV and DSDV, especially the differences caused by on-demand and proactive route queries, via analysis and simulation of the attacks. We explore the potential connections between these properties and the vulnerabilities of the protocols. False distance vector and false destination sequence attacks are studied through simulation. Two connection scenarios: common destination and uniformly distributed traffic load, are examined and the output parameters are measured by varying the number of connections and the maximum speed of node movement. The experiments investigate: (1) The impacts of attacks on network performance, especially the packet delivery ratio. (2) The propagation of false routes, especially the number of innocent nodes that are cheated. (3) The communication overhead to conduct the attacks. (4) The anomaly patterns that can be used to detect the attacks.

The major observations of the analysis and experiments include:

- False destination sequence attacks put more severe impacts on network performance than false distance vector attacks.
- The on-demand route query enables the malicious nodes to conduct real time attacks on the protocols with flexibility.
- The communication overhead of conducting attacks on proactive protocols is independent of the methods of the attacks and the width of the targets.
- A single false route propagates slower in AODV than in DSDV.
- The anomaly patterns of sequence numbers detected by destination nodes can be applied to detect the false destination sequence attacks. The detection of such attacks in AODV heavily depends on the mobility of nodes while in DSDV it is not.

7.1.4 Defending against false destination sequence attacks in AODV

Intruder identification strengthens the protection to mobile ad hoc networks by identifying and isolating the attackers from the rest of the network. We propose a set of criteria: accuracy, overhead, effectiveness, and robustness, to evaluate the identification algorithms. As a practical example, we investigate intruder identification in AODV. Reverse Labeling Restriction (RLR) is proposed to identify and isolate the malicious nodes that attack AODV with false destination sequence numbers. Every node maintains a blacklist to record the providers of the false information detected by it. The labeling procedure is reverse to the propagation path of the information. To prevent network partitions caused by high false positive ratio, the time duration that a suspicious attacker stays in the blacklist increases exponentially to the number of times that it is labeled. The nodes exchange their blacklists to assist the identification of attackers based on indirect experiences. Isolation is achieved by ignoring the routing information provided by the suspicious attackers in the blacklist.

The simulation of RLR is conducted using ns2 and different network scenarios are studied by varying node mobility and the number of independent attackers. The sample experiments include: (1) Study the increase in packet delivery ratio after the attackers are located and isolated. (2) Investigate the communication and computation overhead introduced by RLR. (3) Study the false positive and false negative ratio. The experiments show that most of the innocent nodes can identify all attackers and RLR does not introduce many false positive alarms. Isolating the malicious nodes brings a 30% increase in the data delivery ratio. The robustness analysis shows that RLR does not introduce new vulnerabilities. The analysis also shows that with minor updates, RLR can be combined with other distance vector routing protocols, such as DSDV, for ad hoc networks.

7.2 Future work

Two threads have steered the research on security and privacy in wireless networks and distributed systems: the rigorous foundations for information theory, and innovative applications in pervasive systems. When planning the future research tasks, we consider the potential contributions to both of them. The proposed research includes both continued efforts in securing wireless networks and applications and the exploration of new research areas. Below we sketch three of the tasks.

7.2.1 Effective representation of security data

With the increasing complexity of the security mechanisms, representing the security data effectively and extracting the most useful information from the continuous flow in real time becomes a challenging problem. The driving goal of the research is to establish a platform and develop a series of techniques to represent the security data and facilitate the quick inferences for situational awareness and potential malicious attacks. One example is to continue the research on utilizing visualization tools to detect anomalies in network topology and traffic flows after the information has been conveyed to multi-dimensional multi-variate data sets. We plan to develop innovative mechanisms that attack network security problems from the view of flow database, data mining, computer graphics, and scientific visualization.

7.2.2 Security in wireless mesh networks

Wireless mesh networks have attracted more and more attention from both academic researchers and industry. The hybrid of “static” and “ad hoc” networks offers some advantages over previous approaches. The security issues in such environments put new challenges to the problems such as the design and choice of link level protection and protocol level intrusion detection, information integrity and user privacy, and the fairness of resource usage. We plan to develop innovative security and privacy enforcement mechanisms for mesh networks based on the research in hybrid wireless networks. The problems that are of special interest include the security of routing protocols, key distribution and refreshment, and the detection and revocation of service agreement violators.

7.2.3 Secure pervasive systems

The development of networking technologies and the deployment of new protocols increases the heterogeneity in wireless networks, mobile users, and portable devices. The heterogeneity puts severe challenges to the design of security mechanisms. We plan to continue the research on developing secure and dependable protocols for hybrid wireless networks that explore the assistance from the established infrastructures. The problems that are of special interest include: the negotiation and verification of security requirements among various networks, vulnerability assessment of different protocols and their impacts on the safety of the system, and the design of self-configurable security mechanisms that can adapt to the differences in available bandwidth and processing capability. The research will evolve with the development in pervasive systems.

LIST OF REFERENCES

LIST OF REFERENCES

[1] S. Vasudevan, J. Kurose, and D. Towsley, “On neighbor discovery in wireless networks with directional antennas,” in *Proceedings of IEEE INFOCOM*, 2005.

[2] G. Jakllari, W. Luo, and S. Krishnamurthy, “An integrated neighbor discovery and mac protocol for ad hoc networks using directional antennas,” in *Proceedings of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, 2005.

[3] R. Madan and S. Lall, “An energy-optimal algorithm for neighbor discovery in wireless sensor networks,” in *Proceedings of International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, 2004.

[4] C. Perkins, “Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers,” in *Proceedings of SIGCOMM*, 1994.

[5] C. Perkins and E. Royer, “Ad-hoc on-demand distance vector routing,” in *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*, 1999.

[6] “Network simulator – ns2.” `http://www.isi.edu/nsnam/ns/`.

[7] D. Johnson, D. Maltz, and J. Jetcheva, *DSR: The dynamic source routing protocol for multi-hop wireless ad hoc network*, ch. 5, pp. 139–172. Ad Hoc Networking, Addison-Wesley, 2001.

[8] S. Marti, T. J. Giuli, K. Lai, and M. Baker, “Mitigating routing misbehavior in mobile ad hoc networks,” in *Proceedings of ACM/IEEE International Conference on Mobile Computing and Networking*, 2000.

[9] H. Yang, X. Meng, and S. Lu, “Self-organized network-layer security in mobile ad hoc networks,” in *Proceedings of ACM Workshop on Wireless Security (WiSe)*, 2002.

[10] M. Zapata and N. Asokan, “Securing ad-hoc routing protocols,” in *Proceedings of ACM Workshop on Wireless Security (WiSe)*, 2002.

[11] Y. Hu, D. Johnson, and A. Perrig, “Sead: Secure efficient distance vector routing for mobile wireless ad hoc networks,” in *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*, 2002.

[12] Y. Hu, A. Perrig, and D. Johnson, “Ariadne: A secure on-demand routing protocol for ad hoc networks,” in *Proceedings of ACM MobiCom*, 2002.

[13] A. Perrig, R. Canetti, D. Song, and D. Tygar, “Efficient and secure source authentication for multicast,” in *Proceedings of Network and Distributed System Security Symposium (NDSS)*, 2001.

[14] S. Yi, P. Naldurg, and R. Kravets, "Security-aware ad hoc routing for wireless networks," in *Proceedings of ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2001.

[15] Y. Zhang and W. Lee, "Intrusion detection in wireless ad-hoc networks," in *Proceedings of ACM MobiCom*, 2000.

[16] V. Bharghavan, "Secure wireless LANs," in *Proceedings of ACM Conference on Computers and Communications Security*, 1994.

[17] Z. Zhou and Z. Haas, "Securing ad hoc networks," *IEEE Networks*, vol. 13, no. 6, pp. 24–30, 1999.

[18] P. Albers and O. Camp, "Security in ad hoc network: A general ID architecture enhancing trust based approaches," in *Proceedings of International Conference on Enterprise Information Systems*, 2002.

[19] O. Kachirski and R. Guha, "Intrusion detection using mobile agents in wireless ad hoc networks," in *Proceedings of IEEE Workshop on Knowledge Media Networking (KMN)*, 2002.

[20] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Rushing attacks and defense in wireless ad hoc network routing protocols," in *Proceedings of ACM Workshop on Wireless Security (WiSe)*, 2003.

[21] M. Corner and B. Noble, "Zero-interaction authentication," in *Proceedings of ACM MobiCom*, 2002.

[22] Y. Hu, A. Perrig, and D. Johnson, "Packet leashes: A defense against wormhole attacks in wireless ad hoc networks," in *Proceedings of IEEE INFOCOM*, 2003.

[23] S. Capkun, L. Buttyan, and J. Hubaux, "Sector: Secure tracking of node encounters in multi-hop wireless networks," in *Proceedings of ACM Workshop on Security of Ad Hoc and Sensor Networks*, 2003.

[24] L. Hu and D. Evans, "Using directional antennas to prevent wormhole attacks," in *Proceedings of Network and Distributed System Security Symposium (NDSS)*, 2004.

[25] G. Pei, M. Gerla, X. Hong, and C.-C. Chiang, "A wireless hierarchical routing protocol with group mobility," in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC)*, 1999.

[26] C. Chiang, "Routing in clustered multihop, mobile wireless networks with fading channel," in *Proceedings of IEEE Singapore International Conference on Networks*, 1997.

[27] E. Royer, "Hierarchical routing in ad hoc mobile networks," *Wireless Communication and Mobile Computing*, vol. 2, no. 5, 2002.

[28] C. Perkins, B. Woolf, and S. Alpert, *Mobile IP: Design Principles and Practices*. Printice Hall, 1998.

[29] B. Gleeson, A. Lin, J. Heinanen, G. Armitage, and A. Malis, "A framework for IP based virtual private networks." IETF RFC 2764, 2000.

[30] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP tunnels." IETF RFC 3209, 2001.

[31] S. Bellovin, "Security problems in the TCP/IP protocol suite," *Computer Communications Review*, vol. 19, no. 2, pp. 32–48, 1989.

[32] B. Dahill, B. Levine, E. Royer, and C. Shields, "A secure routing protocol for ad hoc networks," Technical Report 02-32, Department of Computer Science, University of Massachusetts, Amherst, 2001.

[33] P. Papadimitratos and Z. Haas, "Secure routing for mobile ad hoc networks," in *Proceedings of SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS)*, 2002.

[34] Y. Ko, V. Shankarkumar, and N. Vaidya, "Medium access control protocols using directional antennas in ad hoc networks," in *Proceedings of IEEE INFOCOM*, 2000.

[35] R. Choudhury, X. Yang, R. Ramanathan, and N. Vaidya, "Using directional antennas for medium access control in ad hoc networks," in *Proceedings of ACM MobiCom*, 2002.

[36] N. Sastry, U. Shanker, and D. Wagner, "Secure verification of location claims," in *Proceedings of ACM Workshop on Wireless Security (WiSe)*, 2003.

[37] P. Bahl and V. N. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in *Proceedings of IEEE INFOCOM*, 2000.

[38] A. Ward, A. Jones, and A. Hopper, "A new location technique for the active office," *IEEE Personnel Communications*, vol. 4, no. 5, pp. 42–47, 1997.

[39] N. Jain, S. Das, and A. Nasipuri, "A multichannel mac protocol with receiver-based channel selection for multihop wireless networks," in *Proceedings of International Conference on Computer Communications and Networks (IC3N)*, 2001.

[40] E. Jung and N. Vaidya, "A power control mac protocol for ad-hoc networks," in *Proceedings of ACM MobiCom*, 2002.

[41] R. Pickholtz, D. Schilling, and L. Milstein, "Theory of spread spectrum communications – a tutorial," *IEEE Transactions on Communications*, vol. 20, no. 5, pp. 855–884, 1982.

[42] V. Gupta, S. Krishnamurthy, and M. Faloutsos, "Denial of service attacks at the mac layer in wireless ad hoc networks," in *Proceedings of Military Communication Conference (Milcom)*, 2002.

[43] P. Bjorklund, P. Varbrand, and D. Yuan, "Resource optimization of spatial TDMA in ad hoc radio networks: A column generation approach," in *Proceedings of IEEE INFOCOM*, 2003.

[44] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," *ACM Transactions on Programming Languages and Systems*, vol. 4, no. 3, pp. 382–401, 1982.

[45] P. Misra and P. Enge, *Global Positioning System, Signals, Measurements, and Performance*. Ganga-Jamuna Press, 2001.

[46] Y. Ko and N. Vaidya, “Location-aided routing (LAR) in mobile ad hoc networks,” in *Proceedings of ACM MobiCom*, 1998.

[47] S. Basagni, I. Chlamtac, and V. Syrotiuk, “Dynamic source routing for ad hoc networks using the global positioning system,” in *Proceedings of IEEE Wireless Communication and Networking Conference*, 1999.

[48] R. Jain, A. Puri, and R. Sengupta, “Geographical routing using partial information for wireless ad hoc networks,” *IEEE Personal Communication*, pp. 48–57, 2001.

[49] Y. Xu, J. S. Heidemann, and D. Estrin, “Geography-informed energy conservation for ad hoc routing,” in *Proceedings of ACM MobiCom*, 2001.

[50] H. Takagi and L. Kleinrock, “Optimal transmission ranges for randomly distributed packet radio terminals,” *IEEE Transactions on Communications*, vol. 32, no. 3, pp. 246–257, 1984.

[51] T. Hodes and R. Katz, “Composable ad hoc location based services for heterogeneous mobile clients,” *Wireless Networks*, vol. 5, no. 5, pp. 411–427, 1999.

[52] J. Agre, A. Akinyemi, L. Ji, R. Masuoka, and P. Thakkar, “A layered architecture for location-based services in wireless ad hoc networks,” in *Proceedings of IEEE Aerospace Conference*, 2002.

[53] S. Zhu, S. Xu, S. Setia, and S. Jajodia, “LHAP: A lightweight hop-by-hop authentication protocol for ad-hoc networks,” in *Proceedings of International Workshop on Mobile and Wireless Network*, 2003.

[54] A. Applewhite, “What knows where you are?,” *IEEE Pervasive Computing*, Oct-Dec 2002.

[55] “Federal communications commission (FCC) report 03-133,” 2003. `http://hraunfoss.fcc.gov/edocs_public/attachmatch/FCC-03-133A1.pdf`.

[56] D. Mills, “A computer-controlled LORAN-C receiver for precision timekeeping,” Technical Report 92-3-1, Department of Electrical and Computer Engineering, University of Delaware, 1992.

[57] D. Mills, “A precision radio clock for wwv transmissions,” Technical Report 97-8-1, Department of Electrical and Computer Engineering, University of Delaware, 1997.

[58] J. Elson, L. Girod, and D. Estrin, “Fine-grained network time synchronization using reference broadcasts,” in *Proceedings of Symposium on Operating Systems Design and Implementation*, 2002.

[59] M. Sichitiu and C. Veerarittiphan, “Simple, accurate time synchronization for wireless sensor networks,” in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC)*, 2003.

[60] J. Greunen and J. Rabaey, “Lightweight time synchronization for sensor networks,” in *Proceedings of ACM Workshop on Wireless Sensor Networks and Applications*, 2003.

[61] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins, *Global Positioning System: Theory and Practice*. Springer, 1997.

[62] A. Brown, "High accuracy GPS and antijam protection using a p(y) code digital beamsteering receiver," in *Proceedings of ION GPS Conference*, 2001.

[63] L. Scott, "Anti-spoofing and authenticated signal architectures for civil navigation systems," in *Proceedings of ION GPS/GNSS Conference*, 2003.

[64] K. D. Rao, "Anti-FM jamming in GPS receivers using a Kalman-type nonlinear adaptive filter," in *Proceedings of ION GPS/GNSS Conference*, 2003.

[65] P. Kruss, "A survey of multicast security issues and architectures," in *Proceedings of National Information Systems Security Conference*, pp. 408–420, 1998.

[66] M. Brown, D. Cheung, D. Hankerson, J. Hernandez, M. Kirkup, and A. Menezes, "PGP in constrained wireless devices," in *Proceedings of USENIX Security Symposium*, 2000.

[67] N. Courtois, L. Goubin, and J. Patarin, "Flash, a fast multivariate signature algorithm," in *Proceedings of Cryptographers' Track RSA Conference*, 2001.

[68] G. Poupard and J. Stern, "On the fly signatures based on factoring," in *Proceedings of ACM Conference on Computer and Communications Security*, pp. 37–45, 1999.

[69] C. Boyd and A. Mathuria, "Key establishment protocols for secure mobile communications: A selective survey," *Lecture Notes in Computer Science*, vol. 1438, pp. 344–355, 1998.

[70] J. Hubaux, L. Buttyan, and S. Capkun, "The quest for security in mobile ad hoc network," in *Proceedings of ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2001.

[71] B. Shrader, M. Sanchez, and T. Giles, "Throughput-delay analysis of conflict-free scheduling in multihop ad-hoc networks," in *Proceedings of Swedish Workshop on Wireless Ad-hoc Networks*, 2003.

[72] N. Bansal and Z. Liu, "Capacity, delay and mobility in wireless ad-hoc networks," in *Proceedings of IEEE INFOCOM*, 2003.

[73] G. Sharma and R. R. Mazumdar, "Delay and capacity trade-offs for wireless ad hoc networks with random mobility." Technical Report, School of Electrical and Computer Engineering, Purdue University, 2003.

[74] E. Perevalov and R. Blum, "Delay limited capacity of ad hoc networks: Asymptotically optimal transmission and relaying strategy," in *Proceedings of IEEE INFOCOM*, 2003.

[75] C. Perkins, E. Royer, and S. Das, "Performance comparison of two on-demand routing protocols for ad hoc networks," in *Proceedings of IEEE INFOCOM*, 2000.

[76] T. Camp, J. Boleng, B. Williams, L. Wilcox, and W. Navidi, "Performance comparison of two location based routing protocols for ad hoc networks," in *Proceedings of IEEE INFOCOM*, 2002.

[77] J. Kim and M. Krunz, "Fluid analysis of delay performance for QoS support in wireless networks," in *Proceedings of International Conference on Network Protocols (ICNP)*, 1999.

[78] Y. Lu and B. Bhargava, "Self-adjusting congestion avoidance routing protocol for ad hoc networks." Technical Report, Department of Computer Sciences, Purdue University, 2003.

[79] D. Harkins and D. Carrel, "The internet key exchange (IKE) protocol." IETF RFC 2409, 1998.

[80] W. Torgeson, "Multidimensional scaling of similarity," *Psychometrika*, vol. 30, pp. 379–393, 1965.

[81] J. Kong, Z. Ji, W. Wang, M. Gerla, R. Bagrodia, and B. Bhargava, "On wormhole attacks in under-water sensor networks: A two-tier localization approach." Technical Report CS-040051, University of California, Los Angeles, 2004.

[82] M. Davison, *Multidimensional Scaling*. John Wiley and Sons, 1983.

[83] Y. Shang, W. Ruml, Y. Zhang, and M. Fromherz, "Localization from mere connectivity," in *Proceedings of ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2003.

[84] X. Ji and H. Zha, "Sensor positioning in wireless ad-hoc sensor networks with multidimensional scaling," in *Proceedings of IEEE INFOCOM*, 2004.

[85] P. Biswas and Y. Ye, "Semidefinite programming for ad hoc wireless sensor network localization," in *Proceedings of Information Processing in Sensor Networks (IPSN)*, 2004.

[86] G. Xie and J. Gibson, "A networking protocol for underwater acoustic networks." Technical Report TR-CS-00-02, Department of Computer Science, Naval Postgraduate School, 2000.

[87] E. Sozer, M. Stojanovic, and J. Proakis, "Undersea acoustic networks," *IEEE Journal of Oceanic Engineering*, vol. OE-25, no. 1, pp. 72–83, 2000.

[88] J. Proakis, E. Sozer, J. Rice, and M. Stojanovic, "Shallow water acoustic networks," *IEEE Communications Magazine*, pp. 114–119, 2001.

[89] A. Ladd, K. Bekris, A. Rudys, G. Marceau, L. Kavraki, and D. Wallach, "Robotics-based location sensing using wireless Ethernet," in *Proceedings of ACM MobiCom*, 2002.

[90] N. Priyantha, A. Chakraborty, and H. Padmanabhan, "The cricket location support system," in *Proceeding of ACM MobiCom*, 2000.

[91] A. Savvides, C. Han, and M. Srivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *Proceedings of ACM MobiCom*, 2001.

[92] C. Savarese, J. Rabaey, and J. Beutel, "Locationing in distributed ad-hoc wireless sensor networks," in *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2001.

[93] D. Niculescu and B. Nath, "Ad hoc positioning system (APS) using AoA," in *Proceedings of IEEE INFOCOM*, 2003.

[94] C. Savarese, K. Langendoen, and J. Rabaey, "Robust positioning algorithms for distributed ad-hoc wireless sensor networks," in *Proceedings of USENIX Technical Annual Conference*, pp. 317–328, 2001.

[95] K. Rmer, "Time synchronization in ad hoc networks," in *Proceedings of ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2001.

[96] L. Eschenauer and V. Gligor, "A key-management scheme for distributed sensor networks," in *Proceedings of ACM Conference on Computer and Communications Security (CCS)*, 2002.

[97] A. Perrig, H. Chan, and D. Song, "Random key predistribution schemes for sensor networks," in *Proceedings of Symposium on Security and Privacy*, 2003.

[98] R. Pietro, L. Mancini, and A. Mei, "Random key-assignment for secure wireless sensor networks," in *Proceedings of ACM Workshop on Security of Ad Hoc and Sensor Networks*, pp. 62–71, 2003.

[99] W. Du, J. Deng, Y. Han, and P. Varshney, "A pairwise key pre-distribution scheme for wireless sensor networks," in *Proceedings of ACM Conference on Computer and Communications Security (CCS)*, 2003.

[100] D. Liu and P. Ning, "Location-based pairwise key establishments for static sensor networks," in *Proceedings of ACM Workshop on Security of Ad Hoc and Sensor Networks*, pp. 72–82, 2003.

[101] O. Garcia-Panyella, "An easy-to-code smoothing algorithm for 3D reconstructed surfaces," in *Graphics Programming Methods*, pp. 139–146, Charles River Media, Inc., 2003.

[102] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface reconstruction from unorganized points," in *Proceedings of ACM SIGGRAPH*, pp. 71–78, 1992.

[103] G. Golub and C. Loan, *Matrix Computations*. Johns Hopkins University Press, 1996.

[104] A. Kaya and S. Yauchi, "An acoustic communication system for subsea robot," *Oceans*, pp. 765–770, 1989.

[105] D. Kilfoyle and A. Baggeroer, "The state of the art in underwater acoustic telemetry," *IEEE Journal of Oceanic Engineering*, vol. OE-25, no. 1, pp. 4–27, 2000.

[106] C. Coble, M. Elaine, and R. Dale, *Earth Science*. Prentice Hall, 1987.

[107] M. Gross, *Oceanography*. Columbus: Merrill, 1990.

[108] R. Rivest, "The RC5 encryption algorithm," in *Proceedings of International Workshop on Fast Software Encryption*, pp. 86–96, 1994.

[109] R. Baldwin and R. Rivest, "The RC5, RC5-CBC, RC5-CBC-Pad, and RC5-CTS algorithms." IETF RFC 2040, 1996.

[110] B. Kaliski and Y. Yin, "On differential and linear cryptanalysis of the RC5 encryption algorithm," *Lecture Notes in Computer Science*, pp. 171–184, 1995.

[111] P. Levis, S. Madden, D. Gay, J. Polastre, R. Szewczyk, A. Woo, E. Brewer, and D. Culler, "The emergence of networking abstractions and techniques in TinyOS," in *Proceedings of Symposium on Networked Systems Design and Implementation*, 2004.

[112] V. Subramonian, H. Huang, and S. Datar, "Priority scheduling in TinyOS : A case study." Technical Report CSE-TR-74, Washington University, 2003.

[113] C. Karlof, N. Sastry, and D. Wagner, "Tinysec: A link layer security architecture for wireless sensor networks," in *Proceedings of ACM Conference on Embedded Networked Sensor Systems*, 2004.

[114] G. Young and A. Householder, "Discussion of a set of points in terms of their mutual distances," *Psychometrika*, vol. 3, pp. 19–22, 1938.

[115] T. Mitsa and K. J. Parker, "Digital halftoning using a blue-noise mask," in *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, 1991.

[116] W. Wang and B. Bhargava, "Visualization of wormhole attacks in sensor networks," in *Proceedings of ACM Workshop on Wireless Security (WiSe)*, 2004.

[117] M. Duarte and Y. Hu, "Vehicle classification in distributed sensor networks," *Journal of Parallel and Distributed Computing*, vol. 64, no. 7, pp. 826–838, 2004.

[118] V. Park and M. Corson, "A highly adaptable distributed routing algorithm for mobile wireless networks," in *Proceedings of IEEE INFOCOM*, 1997.

[119] Z. Hass and M. Pearlman, "The performance of query control schemes for the zone routing protocol," *IEEE/ACM Transactions on Networking*, vol. 9, pp. 427–438, 2001.

[120] S. Capkun, L. Buttyan, and J.-P. Hubaux, "Self-organized public-key management in ad hoc wireless networks," in *Proceedings of ACM Workshop on Wireless Security (WiSe)*, 2002.

[121] P. Papadimitratos and Z. Haas, "Performance evaluation of secure routing for mobile ad hoc networks," in *Proceedings of ACM Workshop on Wireless Security (WiSe)*, 2002.

[122] P. Sinha, R. Sivakumar, and V. Bharghavan, "Enhancing ad-hoc routing with dynamic virtual infrastructures.," in *Proceedings of IEEE INFOCOM*, 2001.

[123] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens, "An on-demand secure routing protocol resilent to Byzantine failures," in *Proceedings of ACM Workshop on Wireless Security (WiSe)*, 2002.

[124] W. Wang, Y. Lu, and B. Bhargava, "Intruder identification in ad hoc on-demand distance vector protocol." Technical Report, Department of Computer Sciences, Purdue University, 2002.

[125] M. Grossglauser and D. Tse, "Mobility increases the capacity of ad-hoc wireless networks," in *Proceedings of IEEE INFOCOM*, 2001.

[126] B. Bhargava and Y. Zhong, "Authorization based on evidence and trust," in *Proceedings of Data Warehouse and Knowledge Management Conference (DaWak)*, 2002.

[127] M. Corson and A. Ephremides, "A distributed routing algorithm for mobile radio networks," in *Proceedings of Military Communications Conference*, 1989.

[128] E. Royer and C.-K. Toh, "A review of current routing protocols of ad hoc mobile wireless networks," *IEEE Personal Communication*, vol. 6, no. 2, pp. 46–55, 1999.

[129] S. Schneider, "Formal analysis of a non-repudiation protocol," in *Proceedings of IEEE Computer Security Foundations Workshop (CSFW)*, 1998.

[130] J. Zhou and D. Gollmann, "Evidence and non-repudiation," *Journal of Network and Computer Applications*, vol. 20, no. 3, pp. 267–281, 1997.

[131] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical network support for IP traceback," in *Proceedings of ACM SIGCOMM*, 2000.

[132] D. Song and A. Perrig, "Advanced and authenticated marking schemes for IP traceback," in *Proceedings of IEEE INFOCOM*, 2001.

[133] K. Park and H. Lee, "On the effectiveness of probabilistic packet marking for IP traceback under denial of service attack," in *Proceedings of IEEE INFOCOM*, 2001.

[134] G. Kim, T. Bogovic, and D. Chee, "Active edge-tagging (ACT): An intruder identification and isolation scheme in active networks," in *Proceedings of IEEE Symposium on Computers and Communications (ISCC)*, 2001.

[135] D. Balfanz, D. Smetters, P. Stewart, and H. Wong, "Talking to strangers: Authentication in ad-hoc wireless networks," in *Proceedings of Network and Distributed System Security Symposium (NDSS)*, 2002.

[136] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing robust and ubiquitous security support for MANET," in *Proceedings of IEEE International Conference on Network Protocols (ICNP)*, 2001.

[137] D. Dubois, L. Godo, H. Prade, and A. Zapico, "On the possibilistic decision model: From decision under uncertainty to case-based decision," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 7, no. 6, pp. 631–670, 1999.

VITA

VITA

Weichao Wang received the Doctor of Philosophy degree in computer science from Purdue University, West Lafayette in August 2005. He received the master degree in computer science from Purdue University in May 2002. Before joining Purdue University, he received the bachelor (with honor) and master degree in computer science from Tsinghua University in 1998 and 2000, respectively. His research is in the broad area of security and privacy in networks and distributed systems. Special efforts have been put on the design of protocols to improve security and privacy in mobile ad hoc networks, sensor networks, and the emerging applications in pervasive computing.