3-26-2008

Foundational Forensic Techniques for Cellular and Ad Hoc Multi-hop Networks

Xiwei Zhao
Florida International University, xzhao001@fiu.edu

DOI: 10.25148/etd.FI08081554

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

FOUNDATIONAL FORENSIC TECHNIQUES FOR CELLULAR AND AD HOC MULTI-HOP NETWORKS

A dissertation submitted in partial fulfillment of the

requirements for the degree of

DOCTOR OF PHILOSOPHY

in

ELECTRICAL ENGINEERING

by

Xiwei Zhao

2008

To: Interim Dean Amir Mirmiran
College of Engineering and Computing

This dissertation, written by Xiwei Zhao, and entitled Foundational Forensic Techniques for Cellular and Ad Hoc Multi-hop Networks, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

Kang K. Yen

Kia Makki

Hao Zhu

Zesheng Chen

Ronald Giachetti

Niki Pissinou, Major Professor

Date of Defense: March 26, 2008

The dissertation of Xiwei Zhao is approved.

Interim Dean Amir Mirmiran
College of Engineering and Computing

Dean George Walker
University Graduate School

Florida International University, 2008

DEDICATION

I dedicate this dissertation to my parents, Yanming and Binquan, for their unconditional encouragement and support.

ACKNOWLEDGMENTS

I would like to first thank my advisor, Dr. Niki Pissinou, whose guidance and insights have allowed me to finish this research. Also, I would like to express my great appreciation to my dissertation committee: Dr. Kang Yen, Dr. Kia Makki, Dr. Hao Zhu, Dr. Zesheng Chen and Dr. Ronald Giachetti for their challenging questions and kind support.

In addition, I would like to acknowledge the source of financial support for this research: the Department of Electrical and Computer Engineering, and the Telecommunications and Information Technology Institute. Particularly, I would like to thank Dr. Kang Yen, Dr. Niki Pissinou and Dr. Kia Makki who provided me with the opportunities to work for them.

Finally, I want to thank my wife, Rui Song. Without her support, this dissertation would not have been possible.

ABSTRACT OF THE DISSERTATION

FOUNDATIONAL FORENSIC TECHNIQUES FOR CELLULAR AND AD HOC MULTI-HOP NETWORKS

by

Xiwei Zhao

Florida International University, 2008

Miami, Florida

Professor Niki Pissinou, Major Professor

The Internet has become an integral part of our nation’s critical socio-economic infrastructure. With its heightened use and growing complexity however, organizations are at greater risk of cyber crimes. To aid in the investigation of crimes committed on or via the Internet, a network forensics analysis tool pulls together needed digital evidence. It provides a platform for performing deep network analysis by capturing, recording and analyzing network events to find out the source of a security attack or other information security incidents. Existing network forensics work has been mostly focused on the Internet and fixed networks. But the exponential growth and use of wireless technologies, coupled with their unprecedented characteristics, necessitates the development of new network forensic analysis tools.

This dissertation fostered the emergence of a new research field in cellular and ad-hoc network forensics. It was one of the first works to identify this problem and offer fundamental techniques and tools that laid the groundwork for future research. In particular, it introduced novel methods to record network incidents and report logged incidents. For recording incidents, location is considered essential to documenting network incidents. However, in network topology spaces, location cannot be measured due to absence of a ‘distance metric’. Therefore, a novel solution was proposed to label locations of nodes within network topology spaces, and then to authenticate the identity of nodes in ad hoc environments. For reporting logged incidents, a novel technique based on Distributed Hash Tables (DHT) was adopted. Although the direct use of DHTs for reporting logged incidents would result in an uncontrollably recursive traffic, a new mechanism was introduced that overcome this recursive process.

These logging and reporting techniques aided forensics over cellular and ad-hoc networks, which in turn increased their ability to track and trace attacks to their source. These techniques were a starting point for further research and development that would result in equipping future ad hoc networks with forensic components to complement existing security mechanisms.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

FIGURE PAGE

Chapter 1 Introduction

1.1 Background and Motivation

Nowadays, the Internet has become an integral part of our nation's critical socio-economic infrastructure and a progressively dominant role of an individual's life for almost everything from shopping and banking to power grid and defense systems. With its heightened use and growing complexity, the Internet together with other types of computer networks such as cellular networks, sensor networks, or ad hoc networks greatly enriched people's lives; however, the advances in computer networks also energize those with questionable motives to abuse services provided over networks and disrupt social framework. This situation makes network security a critical issue that should be a necessary in designing any types of computer networks.

As a sub-field of network security, Network Forensics is "the next step in the analysis of network attacks, intrusions, and misuses" [Rob06]. During the investigation of crimes committed on via computer networks, a network forensic analyst initially should collects the related evidence remaining after the crimes in order to trace offending actions back and hopefully to locate and identify these perpetrators. Typically, a network forensic analyst starts his work only after a criminal offence has been committed or an alarm has been triggered. Compared to real-time monitoring or strategies i.e. intrusion detection, network forensics is based on evidence left after the completed or attempted crimes and remaining within computer networks.

Intuitively, the more evidence of an unwelcome activity or intrusion enduring in the networks and more information included in the evidence, the greater the chance that the intruder and its misbehaviors may be traced back and identified. Therefore, specialized technologies should be provided to support basic monitoring functions of capturing and recording network activities so that sentinel actions to guard users of computer networks against security breaches and related incidents can be carried out.

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
03/28-11:00:08.909078 131.94.119.3:1985 -> 224.0.0.2:1985
UDP TTL:1 TOS:0xC0 ID:0 IpLen:20 DgmLen:48
Len: 20
00 00 08 01 04 55 64 00 43 45 41 53 00 00 00 00    .....Ud.CEAS....
83 5E 77 01
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
03/28-11:00:09.144240 ARP who-has 192.168.1.99 tell 131.94.119.112
03/28-11:00:09.275474 ARP who-has 192.168.1.1 tell 192.168.1.238
03/28-11:00:09.497014 10.94.119.2:1985 -> 224.0.0.2:1985
UDP TTL:1 TOS:0xC0 ID:0 IpLen:20 DgmLen:48
Len: 20
00 00 10 01 04 6E C8 00 43 45 41 53 00 00 00 00    .....n..CEAS....
0A 5E 77 01
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
03/28-11:00:09.666863 ARP who-has 10.94.119.44 tell 10.94.119.44
03/28-11:00:09.739699 10.94.119.3:1985 -> 224.0.0.2:1985
UDP TTL:1 TOS:0xC0 ID:0 IpLen:20 DgmLen:48
Len: 20
00 00 08 01 04 55 C8 00 43 45 41 53 00 00 00 00    .....U..CEAS....
0A 5E 77 01
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
```

Figure 1-1. An example of log files

As shown in Figure 1-1, ‘data logging’ is the principal and wide accepted technology employed to promptly record network incidents within a certain scope. At every node (or computer host) of a network, a computer program regarding ‘data logging’ runs and keeps recording network incidents occurred within the surrounding area where the node is watching. Due to the overlap of the watching scopes, a network incident may be documented more than once by several nodes that monitored the incident from their own perspectives, and later, forensic analysis can be processed by examining the documents from all these nodes that involved in the incident as participants or as auditors. Furthermore, by collecting, inspecting and analyzing all documents from ‘data logging’ throughout the network, any incident occurred within the scope of the whole network can always be traced.

Unfortunately, direct approaches always leave a lot to be desired, because the technology of ‘data logging’ may be challenged or disrupted, and also because the documents generated from the process of ‘data logging’ (or log files) may be distorted or destructed. Specifically, criminals or hackers often hide their trails or spoof their identities before initiating an intrusion or a widespread disruption. Even worse, an accomplished hacker may simple erase all or parts of log files related to his offending actions after his successful intrusion into computers or computer networks.

In order to clearly illustrate challenges that network forensics would face, a typical example of network offenses via the Internet is presented in detail. First, a hacker turns on a computer with access to the Internet. Through this computer, the hacker logs in at a public server as a normal user. This public server is generally called a stepping stone of the following perpetrations, and in this way the hacker disguises the ID of his current computer by hiding himself behind the public server. After that, the hacker attempts to search and locate security holes on a target computer by issuing probing packets from the public server. Mostly, the probing packets are a series of applications for anonymous services provided by the target computer. In case a security hole of *buffer overflow* has been located on the target computer, the hacker launches an intrusion by injecting a piece of executable code into the network service with the security hole. Moreover, if the network service had special privileges, the hacker could then obtain super-user privilege on the target computer, and a successful intrusion has been completed.

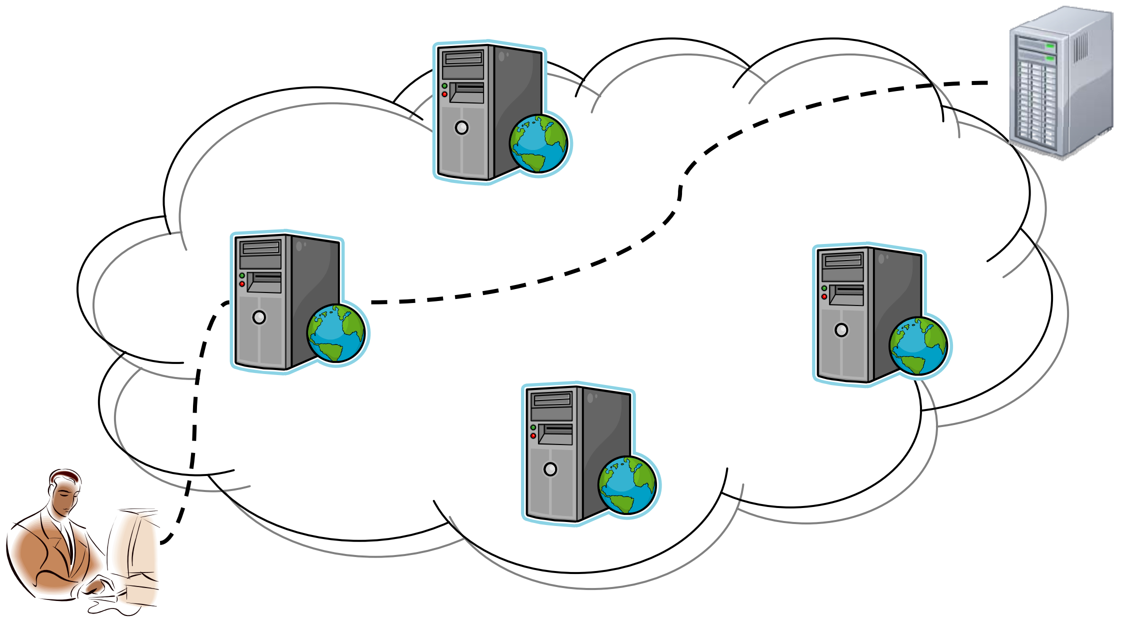

Figure 1-2. A typical intrusion over the Internet

Actually, based on log files of nodes within a computer network, completed or attempted network crimes can always be traced. For the example above, the procedure of this network intrusion is documented through the mechanism of 'data logging' at nodes involved in. On the public server, the *login* transaction of the hacker would be recorded, as are the actions of sending probing packets out. On the target computer, the hacker's probations (or frequent service requests from the public server) are logged, as is the *exception* of *buffer overflow*. Therefore, an investigation of this intrusion can start from the target computer. By analyzing log files maintained at the target computer, the stepping stone of the intrusion, i.e. that public server, would be located. Furthermore, with the log files from the stepping stone collected and analyzed, the hacker's computer may be spotted afterward.

However, some log files may be unavailable during the investigation to trace the intrusion back. In the typical network intrusion illustrated above, the hacker may become a super-user and fully control the target computer. Subsequently, the hacker would tamper with or simply remove the log files maintained at the target computer. As a result, the investigation can not go any further, since the investigator can not locate that public server (i.e. the stepping stone of the intrusion,) without indication from these destructed log files, let alone the computer of the hacker. Even through the evidence of the intrusion was documented at the public server, it has not been reported or published and no other nodes can associate these documents at the public server with the tragedy occurred at the target computer. Therefore, the investigator has no way to know that there still exists evidence of the intrusion at the public server. Moreover, since there are thousands of public servers scattered across the Internet, a wild search throughout all of these public servers is hardly conductive and infeasible.

In addition, the example of network intrusions discussed above is via the Internet which falls into the category of fixed networks. In mobile networks, the challenges of network forensics get even more serious. If the network intrusion is launched over a mobile network, especially a mobile ad hoc network, additional issues regarding network forensics must be taken into account. Within a mobile network, each node or computer, no matter whether it is a hacker or a normal node, would move arbitrarily without a fixed address. As a result, hackers can attack one computer at one place and then move to another place to avoid being traced. Also, nodes may join or leave an ad hoc network arbitrarily. As a result, a hacker may join the network with an identifier, and misbehave all the time. After that, the hacker simply leaves the network, and then joins the network again with another identifier to escape punishment. Therefore, old principles and techniques behind forensics for fixed networks cannot be employed for mobile networks as they do not have the underlying infrastructure older protocols so comfortably assumed to have existed.

In summary, the network forensics is primarily based on log files. With these log files a forensic analysis can be conducted; and with more log files or more details of incidents included in log files, the forensic analysis can then be processed more smoothly. This leads to the first major concern in the research of developing new tools for network forensic analysis: that is, how to record network incidents effectively so that more details of incidents are documented. Additionally, log files within a network are distributed and separately maintained at each node. Thus, collecting wanted information from these distributed log files is still a challenge. To avoid collecting log files blindly, all these distributed log files should report their existence and their contents publically so that the process of collecting information throughout the network can be guided. This also leads to the second major concern in the research of developing new tools for network forensic analysis: that is, how to report and organize the contents of log files efficiently.

1.2 Statement of the Problem

At the end of the previous section, two major concerns of developing new tools for network forensic analysis have been set forth. They can be summarized as the concern of "incident logging," and the concern of "log reporting." In the next sub-sections, these two concerns are analyzed, and the central problems of this research are specified.

1.2.1 Incident Logging

From the perspective of incident logging, it is necessary to list the components that log files should consist of. As previously discussed, network incidents must be logged in advance in order to support later forensics analyses; moreover, the details of these network incidents should be fully documented.

Generally, a network incident includes one or several raw network incidents. In this dissertation, a raw network incident refers to a packet delivery described as "at a time and a place, a node has (or has not) received (or sent) a packet." For example, there are two nodes, *A* and *B*. Node *A* is going to request a file from node *B*. Initially, node *A* sends a request packet to node *B*. Later, node *B* sends back a reply packet indicating that the file is not available. From the view of node *A*, this network incident, i.e. the incident of a failed file request, includes two raw network incidents that occur successively:

1. At T_1 time, and P_1 place, node *A* has sent a request packet.
2. At T_2 time ($T_2 > T_1$), and P_2 place, node *A* has received a reply packet.

By these two raw incidents, the incident of that failed file request has been fully described. For these raw incidents, five components are always included. They are *time*, *location*, *node ID*, *action* (receive or send), and *packet*. In fact, these five components are essential to document any incident. With these five components, information related to 'when', 'where', 'who', 'what' of incidents is fully recorded.

In this research, the component of *location* is considered, with regards to its measurement and expression. Generally, a *location* can always be identified by the distances from some landmarks. For instance, the location of a student can be described as 200 yards from the library, 300 yards from the cafeteria and 300 yards from the department of CS. In fact, the distance and distance measurement are essential for the localization. Mathematically, *location* is a concept within normed vector spaces [Kol99] in which a norm (or a *distance metric*) should be given. However, a network topology space has its *distance metric* absent. This implies a problem: that is, without a *distance metric*, the *location*s of nodes can not be measured within a network topology space.

Currently, most network designs introduce a *distance metric* before localizing nodes in network topology spaces. Exactly, those designs require additional devices to measure the distance or proximity of any two nodes. For example, some designs utilize Global Positioning System (GPS) to localize nodes, and require each node to be equipped with a GPS receiver. In fact, these GPS-based designs just embed network topology spaces into 3-dimensional Euclid spaces (or real world). As a result, node localization within the network must depend on the GPS system which runs outside of the network. Even more, the GPS receivers equipped at nodes also incurs extra expenses. Other designs utilize the strength of received signals to estimate the distance that the signal has been transmitted, and then to measure the position of nodes. For these signal strength based designs, node localization relies on the information that comes from the physical layer, and requires physical measurement.

In this dissertation, the problem of node localization is considered. The possible solutions of the problem would work in network topology spaces, without any additional device or physical measurement required. Most importantly, the possible solutions are sufficient to express the *location*s of nodes, for supporting the process of incident logging and the later process of network forensic analysis. In terms of network traceability, it is not necessary to localize the physical location of each node. Within a network, the information of relative positions of these nodes with each other is much important, because connections between nodes are the crucial issue for communications, rather than physical locations of nodes.

1.2.2 Log Reporting

From the perspective of log reporting, all publically reported logs should be organized as a distributed database. In this way, log files an investigator is interested in would be searched and collected conveniently in order to process the forensic analysis over network. As explained in Section 1.1, the attempt to collect all log files from all nodes is quite formidable. To collect log files efficiently, all distributed log files must be publically reported and sorted before the forensic analysis is conducted.

Currently, Distributed Hash Table (DHT) is a popular technology to publish and organize distributed resources. Many DHT designs are proposed, such as CAN [Rat01], Chord [Sto01], Pastry [Row01, May02], Tapestry [Zha03], etc. Likewise, log files among different nodes can also be viewed as distributed resources, and can subsequently be published and organized by following these DHT designs.

However, DHT cannot be directly deployed to publish (or publically report) log files. According to the DHT designs, a node should publish its log files (or resources) by sending out a special packet (called a publishing packet) to a pre-selected node that is determined by the DHT architecture. Nevertheless, delivering the publishing packet over the network is also a new (raw) network incident, and should be logged at intermediate (or relay) nodes along the packet's delivery path. This new log incurred by the publishing packet delivery still needs to be published, and another publishing packet corresponding to this new log would be issued subsequently. As a result, the publishing process will be invoked recursively and endlessly, and publishing packets will finally jam the whole network.

In this dissertation, the problem of this recursive process is considered. The possible solution should stop these recursive processes, and the overhead cost of the solution should be reasonable. In addition, the communication cost of the publishing process also needs to be lessened. In summary, this research incorporates two specified problems, which are 1) localizing nodes in network topology spaces without *distance metric*; and 2) publically reporting and organizing these distributed log files economically.

1.3 The Dissertation Goal

The goal of this research is to establish a framework to support network forensic analysis over mobile networks. This framework includes two parts.

1) The first part focuses on designing the mechanism of incident logging, which runs at different nodes but generates consistent log files. Specifically, the information included in these log files should meet the requirements of a general forensic analysis.

2) The second part addresses the problems of publically reporting these log files. It should be noted that these log files are distributed among different nodes. Specifically,

the outputs of the reporting must cooperate with any potential search regarding an actual forensic analysis.

The purpose of this study is to create tools and algorithms, with regards to supporting forensic analysis over mobile networks. The tools and algorithms should not only support, but also facilitate forensic activities at a reasonable cost under ad hoc or infrastructure based circumstances.

1.4 Significance and Originality

Node localization is examined in this dissertation in order to accurately document where network incidents occurred, as is log reporting aimed at tracing network incidents conveniently. Compared with previous work, a framework in the research of supporting network forensic analysis is founded, and all current designs regarding network forensics can be incorporated under this framework. This research is essential to support network forensics, network traceability or network accountability over mobile networks, as well as to strengthen the security of mobile networks.

Most importantly, a framework for supporting network forensics is initiated. Under the framework, some crucial problems are also inspected and solved.

1.5 Methodology

Based on modern technologies and research models in mobile networks, a theoretical framework has been established. However, because of the wide coverage of establishing this framework, research issues from the different fields (such as Authentication, Distributed Hash Table, etc.) must be taken into account. Moreover, due to the differences of these research issues, research methods are different from case to case. In this section, a general research path of this work is briefly described.

This research utilizes multiple methods that are popular in the field of network design, such as modeling, theoretical analysis, theoretical calculation, mechanism design, and simulation.

For the modeling, the research object (i.e. mobile networks) is formulated, in relation to the research problem concerned. Some key aspects of the object should be emphasized in the model, and some other aspects of the object should be ignored in order to simplify the model.

After the modeling are theoretical analyses and calculations. In the phase of theoretical analysis, the research problems are inspected thoroughly; however, it would be difficult for the theoretical analysis to be completed with a general solution (due to the dynamic topology and other complexities of mobile networks). Therefore, the analysis is mainly devoted to seeking the solutions of the worst scenarios. In addition, the theoretical analysis is aided by theoretical calculation. After the theoretical analysis and calculation, rules and criteria are provided. Those rules and criteria can be utilized to guide network design over mobile networks.

Lastly, simulations are conducted to empirically evaluate the result of theoretical analyses. The algorithms and mechanisms are implemented or customized in network simulators. The objective of simulations is also to measure the performance of the designed mechanisms.

1.6 Targeted Applications

All results of this research can be used for logging network incidents and reporting log files over mobile networks. For logging incident, these documented incidents can also be explored to examine node’s identity. For reporting log, the optimized reporting process can also be applied to maintain distributed databases.

In addition, Location Management system of cellular networks has been extended to track the movement of mobile subscribers, especially to quickly trace the movement of criminal subscribers. Moreover, the system of Location Management has been optimized in order to reduce the cost of *location* registration. Also, technologies of in-network processing are studied in wireless sensor networks, and innovative mechanisms are designed to decrease the cost of reporting physical incidents observed by sensor nodes.

Overall, results of this research mainly developed over mobile networks can also be applied to other network circumstances, since mobile networks are only considered as a research model rather than as actual networks.

1.7 Scope of the Study

In this dissertation, the issues of supporting network forensic analysis are researched over mobile networks. As presented previously, log files are principal materials on which a network forensic analyst works. Therefore, issues related to log files are thoroughly studied. Briefly, this study is extended to two aspects: 1). inside log files, how to generate effective log files is examined; 2). outside log files, how to efficiently report log files is examined.

In respect to generating log files, two components of log files are considered. These components are: *time* and *location*, which identify “when” and “where” the network incident occurs. In the dissertation, the discussion of *location* is covered, and the discussion of *time* can be found in [Zha07].

In respect to reporting log file, the emphasis focuses on improving current P2P technologies of resource publishing. Our improvements implant DHT technology onto log reporting over mobile networks. Also, the improvements optimize the process of *location* registration as well as movement tracing in cellular mobile networks. Again, the problem of reporting physical incident is researched in wireless sensor networks.

1.8 Organization of the Dissertation

The remaining chapters are organized as follows: In Chapter 2, an overview covers the research work related to issues that would be further discussed and extended in the following chapters. After the related work, the research efforts of supporting network forensic analysis focuses on the two major concerns, which are incident logging and incident reporting (or log reporting).

First, the incident logging is taken into account, because it is the starting point of supporting network forensics. In Chapter 3, one key component of incident logging, i.e.

location, is discussed. The chapter introduces ‘Background’ to label nodes’ *location*s. In terms of forensic analysis, ‘Background’ is sufficient to be an alternative of the component of *location* in log files. More important, Background does not require a *distance metric*. In the chapter, ‘Background’ is even applied to the process of node authentication.

The other major concern, i.e. log (or incident) reporting, is considered in Chapter 4. The issues of incident reporting are discussed in a general model of IP-based mobile networks. In the chapter, distributed log files are organized under a typical DHT architecture. Moreover, a mechanism has been designed to limit the communication cost of incident reporting.

After the discussion over a general reporting process, the study is specified and concentrated on the most popular mobile networks, i.e. cellular mobile networks. In Chapter 5, the discussion is dedicated to *location* reporting (or registration), and aimed at tracing the criminal’s movement quickly (which is a specific forensic activity) in cellular mobile networks. Meanwhile, the cumulative cost of *location* registration is optimized, in order to obtain better performance.

In Chapter 6, the context of the study changes to wireless sensor networks, which is also popular networks of the current research. Especially, wireless sensor networks can be used to monitor physical objects, and report physical incidents which are much helpful on network forensics. In the chapter, the technology of “in-networking process” is extended.

According to the extension, physical signal fields are explored to report incidents (or sensory data) to sink nodes of wireless sensor networks. In this way, the communication cost of the reporting process would be reduced.

Finally, in Chapter 7, a summary of the research is presented and a prospective of future work is considered.

Chapter 2 Related Work

Current research of network forensic support, regardless of whether it is on fixed networks or on mobile networks, often focuses on defending against some particular types of network attacks, such as denial-of-service attacks, stepping stone attacks [Aad04], identity spoof [Kuz03], jellyfish attacks [Lee01], shrew attacks [Pax01], etc.. Since these solutions and research models are aimed at one or two attack types, solutions and research models can be quite different from each other. In this chapter, the different solutions and research efforts from different fields are covered, in order to comprehensively present the current research status of network forensic support.

As was discussed in the previous chapter, log files are the principal materials of network forensics. Research work on forensic support services should still be related to generating, storing, sorting, publishing, and delivering log files. This research focuses on the processes of generating and publishing log files. In the following discussions, these two processes are formatted as incident logging and log publishing. In this chapter, recent developments and ongoing work about incident logging and log publishing are reviewed and thoroughly examined. This is essential for any network design towards forensic support services.

This chapter includes two sections: “incident logging” and “log publishing.” In the section of incident logging, the *location* of a node and its related applications are reviewed. These related applications include the issues of node localization, identification and authentication. By following the discussion of Chapter 1, the *location* of a node, a crucial component of logging network incidents, can authenticate the node's identity. Thereafter, current authentication designs over mobile networks are summarized.

In the next section, log (or incident) publishing is reviewed under three circumstances. First, research efforts on an IP-based network are concerned, and results about log publishing and attack tracking are presented. Second, the scope of the review narrows down to cellular networks. In this circumstance, the designs of *location* publishing (or *location* registration) are fully inspected. Third, the circumstance changes to wireless sensor networks (WSNs), and the latest innovative designs with regard to incident publishing (or sensory data publishing) are examined.

2.1 Incident Logging

The *location* of a node is a crucial component to accurately document network incidents, and can also be used for the node's authentication. In this section, modern technologies of node localization, identification and authentication are presented.

Authentication is essential for secure communications, and can be achieved by checking something that is related to a node's identity. Currently, the work concerning network authentication depends on key and key management [Mer07], such as a key's generation, exchange, and distribution. For the designs of key-based authentications, two scenarios are discussed here.

The first scenario relies on a third party authority, which is called certification authority (CA). Inside a network, a public key infrastructure is used for data encryption, and CA is in charge of public keys distribution. CA can be a centralized system [Zho99, Hua03] or a distributed system [Go05, Pir04a, Pir04b, Kon01]. In general, these authorities reside within the network. However, these authority nodes must be predefined by some privileges which other common nodes do not possess.

In the second scenario, no authority exists. A client and a server establish their secure connection through negotiation [Tse07, Cap06, Pie06, Mer05, Cap03, Put03, Hub01, Ven00]. In fact, the second scenario is a special case of the first scenario, because each node acts as the authority of itself. However, since each node is the authority of itself, the key of each node can then be assigned or selected by the node itself. As a result, malicious nodes could change their keys (or identities) arbitrarily without any expense.

Key-based authentication builds an association between a node's identity and its assigned key. The assigned key is a man-made identifier (or characteristic) of the node, as is the case with many other characteristics the node possesses. Based on this key, the procedure of authentication is conducted. The node's identity can be confirmed by checking whether its key is correct. In fact, the node's identity can also be confirmed by checking any other characteristic of the node, only if the value of this characteristic is unique to the node. In other words, key-based designs are no longer irreplaceable for the node authentication. Here, the argument is made that a node's neighbor list, called a node's Background, is another feasible choice for node authentication. In addition, the list of neighbors, which is a physical characteristic of the node, can be used for many other applications, such as incident logging and node identification.

Accordingly, the authentication strategies can then be classified into two categories. The first category includes all these key-based authentication strategies in which an authority is always needed, regardless of whether the authority is a central system or a distributed system. The strategy designed in this research belongs to the second category, in which authorities and keys are not necessary. The identity of the node can be confirmed by other characteristics of the node, especially by the node's physical characteristics, e.g. neighbor list. Compared to other characteristics of the node, the assigned key is a characteristic that is controlled by an authority. Conversely, the neighbor list of the node is a physical characteristic that no node can fully control.

2.2 Log Publishing

2.2.1 Log Publishing over General Computer Networks

Currently, with respect to supporting network forensics, all technologies can be grouped into two categories, which are termed "packet marking" and "packet logging". For packet marking, a passing packet may be marked with the node's ID and a time stamp. Also, the node may log each passing packet along with a time stamp, which is called packet logging. In some designs, technologies of packet logging and packet marking are even combined [Gon05].

For those designs of packet marking [Sav00, Bel03], the marked packets will finally be logged at their termination nodes. Thus, both packet marking and packet logging can be described uniformly: at a node, a message is generated when a packet arrived. This message should include information about the packet's ID, the node's ID, and the arrival time. For packet logging, the message is logged locally. For packet marking, the message should be delivered along with the packet, and logged at the packet's termination, or it can be said that all packet-related logs are published at the packet's termination. The difference between packet logging and packet marking is that for packet logging, the logs of a packet arrival are maintained locally; while for packet marking, the logs of a packet arrival are delivered to or published at the termination of the packet.

2.2.1.1 Packet Marking

In fixed networks, it is common that every second, Gigabytes are delivered over backbone links. So, it is impossible for routers along the backbone links to mark every passing packet. Besides, a flood attack implies that lots of attack packets will hit the victim node along the same routing path, and it is not necessary to mark every attack packet at every intermediate node along the attacking path. This situation leads to the strategy of probabilistic packet marking (PPM) [Sav00], according to which each router only marks passing packets with a low probability. By following PPM, a packet may be marked at one or two routers along its delivery path. At the victim node (or the termination node of the packet), the entire delivery path can be reconstructed by collecting marks from a group of these partially marked packets.

A similar idea is also proposed in ICMP trace-back (ITRACE) [Bel03], in which the mark field is split from the passing packets and carried by ICMP messages. According to ITRACE, when a packet arrives at a router, the router will send an ICMP message out with a low probability. In addition, the termination of this ICMP message should be the same as that of the packet. The delivery path of the packet can then be reconstructed at the termination node by identifying the source nodes of these ICMP messages.

As is the case of marking packets with low probability, some designs use other data compression technologies to record the delivery path on packets. In [Dea02], the mark field of each packet is defined with the fixed length, no matter how long the delivery path is. The delivery path that the packet passed is compressed hop by hop, and recorded in the mark field of the packet. Still, the delivery path of the packet can be recovered by collecting marks from numerous packets.

In other designs, the marks of packets are used for identifying IP spoof before the packets reach their terminations [Par01]. Whenever a packet arrives at a node, its source address, termination address, and mark field are examined by the node. If the source address, the termination address, the marks, and the address of the node itself are along a possible routing path from the source to the termination, then the packet is valid. Otherwise, the packet is invalid.

2.2.1.2 Packet Logging

Packet logging requires that all passing packets should be logged locally. Because of the storage limits and the heavy traffic load, each packet should be hashed before being logged at a node. The typical design of packet logging is Source Path Isolation Engine (SPIE) [Sno02], in which a Bloom Filter is used for packet hashing. According to SPIE, all passing packets should be logged at the node where they arrived. A Bloom Filter is deployed for packet hashing, and the output for each packet is a 4 byte data, called packet digest. As it is only 4 bytes long, the collision probability of these packet digests is high. Therefore, SPIE only keeps recent packet digests, and drops older ones, by using the technique of slide window.

To trace the delivery path of a packet, the termination node of the packet should broadcast first. Any node that possesses the related packet digests should reply and initiate new broadcasting. This process would be iterated until the source of the packet is reached. In the paper [Sno02], the authors argue that the first 28 bytes of each packet (IP header plus 8 byte payload) are sufficient to differentiate packets. Therefore, only the first 28 bytes of each packet are fed in the Bloom Filter. In [Mit02], the passing packets are classified first by their source and termination address, before hashing and logging. Additionally, different sliding windows are assigned to each packet class to identify "recent" packets.

Some packet logging strategies go even further on log compression. In their designs, only some traffic patterns of each link, such as traffic speeds, average packet size, or traffic spectrum, are sampled and logged. During the phase of attack tracking, the correlations of these logged traffic patterns over different links are examined, and the most correlated links are picked to reconstruct the attacking path.

In [Hus06], the dominant frequency of the traffic is selected as its fingerprint. In [Sek04], routers record traffic loads of links. While an attack tracking, the correlations between the traffic loads of the current attack and of those records are calculated. The attacking path can then be identified by connecting these links with the highest correlations.

A similar strategy also appears in [Xie05]. According to the paper, the path of worm propagation spans a graph over a network, and this propagation graph can be identified by the strategy of "moonwalk," which is similar to the trace-back procedure of SPIE. At the beginning, the origin graph is a link of the victim node. At each moonwalk step, all hosts that contact the graph will have their traffic logs analyzed in terms of causality and correlation. The most prospective links, which are with high-level causality and correlation, are added to the graph. In [Sek06], the traffic loads are characterized by multiple resolutions to obtain better performance.

In all above designs of packet marking or packet logging, no information has been recorded explicitly to indicate the relative position where the node is along the delivery path of packets. This is not a problem for fixed networks, as all nodes are fixed at their positions with fixed links. However, since nodes can move arbitrarily in mobile networks, the relative position of the node should be logged, along with the time stamp for the later investigation.

In [Hua05], SPIE is extended, and the TTL (time to live) field of each packet is explored. In addition to logging the packet digests (through Bloom filter), the design requests nodes to log the TTL field of each packet separately. As the TTL value of a packet decreases hop by hop along the delivery path, the sequence order of nodes along the packet's delivery path can then be recovered by sorting these logged TTL values.

However, problems still exist when working with mobile networks. For designs of packet logging, all logs are distributed in the network. Hence, it will be difficult to search logs that relate to a particular packet delivery. Some strategies, for example of SPIE, leverage the static topography of fixed networks, and search logs along the backward delivery path hop by hop. In mobile networks, the dynamic topography makes a SPIE searching impossible. Efforts of collecting all logs from all nodes can not solve the problem, not only because the expense of the log collection is massive, but also because the information that is interested would be buried under a multitude of unrelated logs.

For the designs of packet marking, logs related to a particular packet delivery are forwarded (or published) to the packet's termination, which seems much easier for the log collection and the forensic analysis. Nevertheless, the termination node is more likely to be a victim under attacks. As a result, the termination nodes are more likely to be compromised, and log files of the termination nodes would be tampered with or even erased thoroughly. Thus, the results of forensic analysis that solely rely on logs from the victim nodes are questionable.

2.2.2 Location Publishing over Cellular Networks

Over a GSM network, the current *location* of a mobile subscriber can always be traced by referencing its HLR (Home Location Register). According to the standard IS-41, the serving VLR should publish its *location* on the mobile subscriber's HLR whenever the subscriber come in. From the viewpoint of a mobile subscriber, its HLR is fixed at somewhere inside the GSM network, and its VLRs (Visitor Location Register) are scattered in the network. When the subscriber moves into a new *location*, it should register on the local VLR first. After the registration, the local VLR gets to know where the subscriber's HLR is, and then publishes (registers, updates) the subscriber's current *location* on the HLR. Thereafter, the movement tracking of the mobile subscriber can be conducted by simply referencing the log files of the subscriber's HLR.

However, the process of *location* publishing would be costly if the subscriber moves far away from its HLR and changes its *location* frequently. For the movement tracking, it would seem a waste that the investigator keeps referencing the HLR continuously. Many innovative strategies have been developed to reduce cost as well as to support movement tracking over GSM networks. These strategies are focused on optimizing the procedure of *location* publishing.

In the mid-1990s, the concept "anchor" was introduced to the procedure of *location* publishing (registration), aimed at reducing the accumulative expense along the moving path. From the perspective of a subscriber, any of its VLRs can be selected as its anchor only if the selection can reduce the accumulative expense of its *location* publishing.

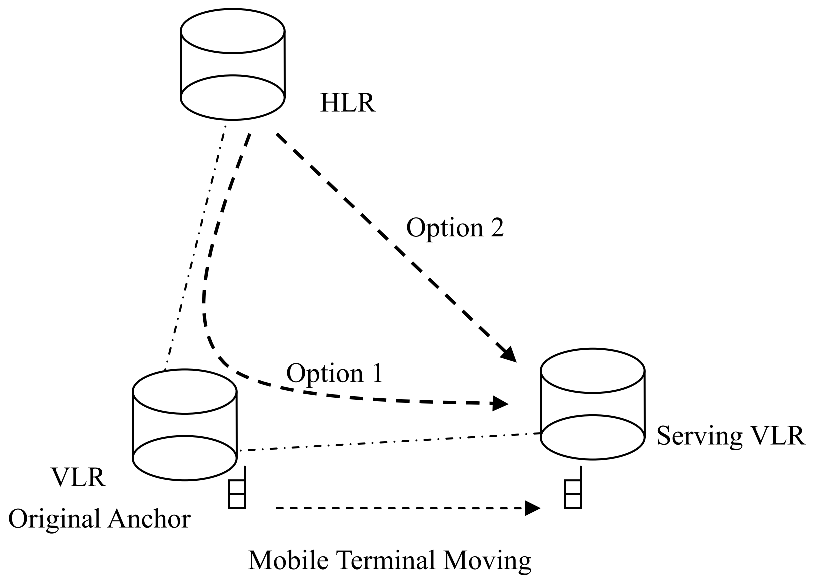

Figure 2-1. Anchor and its update

When roaming, the subscriber only needs to publish its *location* on its anchor, instead of on its HLR. In [Ho95], an anchor is inserted between HLR and the serving VLR. This design partially follows the standard IS-41 when a call arrives at the subscriber. When the subscriber receives a call, the current *location* (indicated by the routing number) of the serving VLR will publish on its HLR directly, and the serving

VLR becomes the anchor of the subscriber (the VLR and MSC are exchangeable in this instance). When the subscriber moves into a new *location*, there are two options to update its anchor. As shown in Figure 2-1, one option is to select the new serving VLR as the anchor. The other option is to keep the original anchor without change. The new serving VLR decides which anchor to select, by taking the current publishing/binding expense and call arrival rate into account.

A similar design has also been developed in the circumstance of inter-network roaming [Wan01]. In [Kal99, Vee97], routing tables are also referenced for the anchor update. In [Li04], the subscriber’s anchor should be selected from other subscribers’ HLRs. In [Rat00], the concepts of VLR and HLR are mixed together and called “Location Register (LR).” In this way, regardless of whether it is HLR or VLR, any one (LR) can be selected as a subscriber’s anchor. Above all, strategies only select one anchor for each mobile subscriber.

Other techniques, such as “terminal caching” and “built-in timer” are also combined with the technology of anchor [Hai01, Hai02, Zho01], regarding the subscriber’s *location* publishing. In these designs, a subscriber is enhanced with a built-in memory [Hai01, Hai02] to cache the address of its anchor. When the subscriber moves to a new *location* served by a new VLR, this new VLR queries the subscriber about the address of its anchor and then binds to the anchor. A built-in timer is also used for optimizing the procedure of *location* publishing [Zho01]. Whenever a built-in timer exceeds its (pre-defined) threshold, the subscriber will initiate a *location* publishing, regardless of whether it has moved or not since the last *location* publishing. With the threshold of the timer carefully defined, the publishing expense could be reduced significantly with an acceptable probability of call blocking and movement tracking.

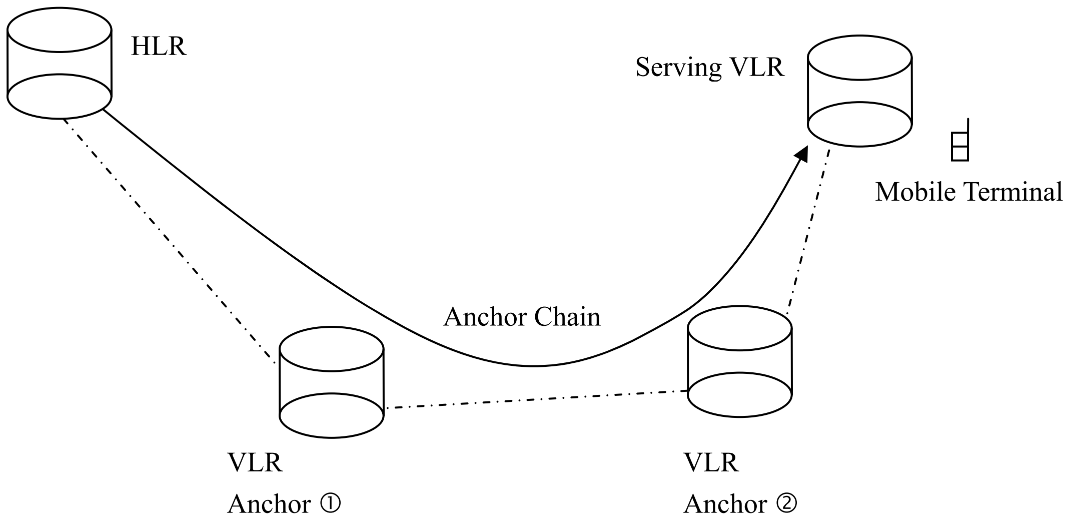

Figure 2-2. Anchor chain

As shown in Figure 2-2, more anchors are introduced in the process of *location* publishing [Bej03], and all these anchors are connected to form a chain. In this way, the movement tracking would be conducted by searching along the chain. Anchors are selected from the VLRs that the mobile subscriber visited, and updated based on the distance between the pair of neighbor anchors. In the worst cases, this design can guarantee that a subscriber's accumulative expense of *location* publishing/binding be with the order of O($M \cdot \log M$). Here, M is the distance that the subscriber has moved. In [Mao04], 2-tier architecture of HLR-VLR has been extended to 3-tier architecture, in which a new tier is inserted between HLR and VLR. In fact, this new tier simply functions as an anchor. Likewise, more tiers are even defined inside the mobile IP network [Ma04] to reduce the expense of *location* publishing.

Overall, the expense of *location* publishing can be reduced by inserting anchors between HLR and the subscriber's serving VLR. Meanwhile, the moving path of a subscriber can still be traced by referencing its HLR and all its anchors. In this dissertation, the author follows the design of anchor chain proposed by [Bej03], and develops an optimization criterion of chain update. In the worst cases, the subscriber's accumulative expense of *location* publishing is with the order of O(M). Here, M is still the distance the subscriber has moved.

2.2.3 Physical Incident Publishing over Wireless Sensor Networks

Wireless Sensor Networks (WSNs) consist of spatially distributed sensor nodes. By collecting sensory data from these distributed sensor nodes, WSNs can monitor physical conditions (i.e. sound, motion, temperature, and pollutants) at different *location*s automatically. In general, the gateway node of a WSN is called a sink node. Sensor nodes with sensory data are called source nodes. From the perspective of source nodes, the sensory data, implying the occurrence of incidents, should be published on the sink nodes. Thus, the data collection of WSNs is a variation of incident publishing.

2.2.3.1 WSN Clustering

To gain energy efficiency, sensory data should be aggregated locally. This is called in-network processing. Mostly, source nodes are far away from the sink node, so it would be costly if sensory data from source nodes are delivered to the sink node individually. In-network processing is to drive these sensory data to converge to some local nodes and aggregate these sensory data there. Only aggregated data would be sent to the sink node. In this way, the total energy expense of data delivery (or incident publishing) may drop substantially. As shown in Figure 2-3, sensory data are converged to node *A*, and aggregated there. The aggregated data would then be delivered to the sink node directly.

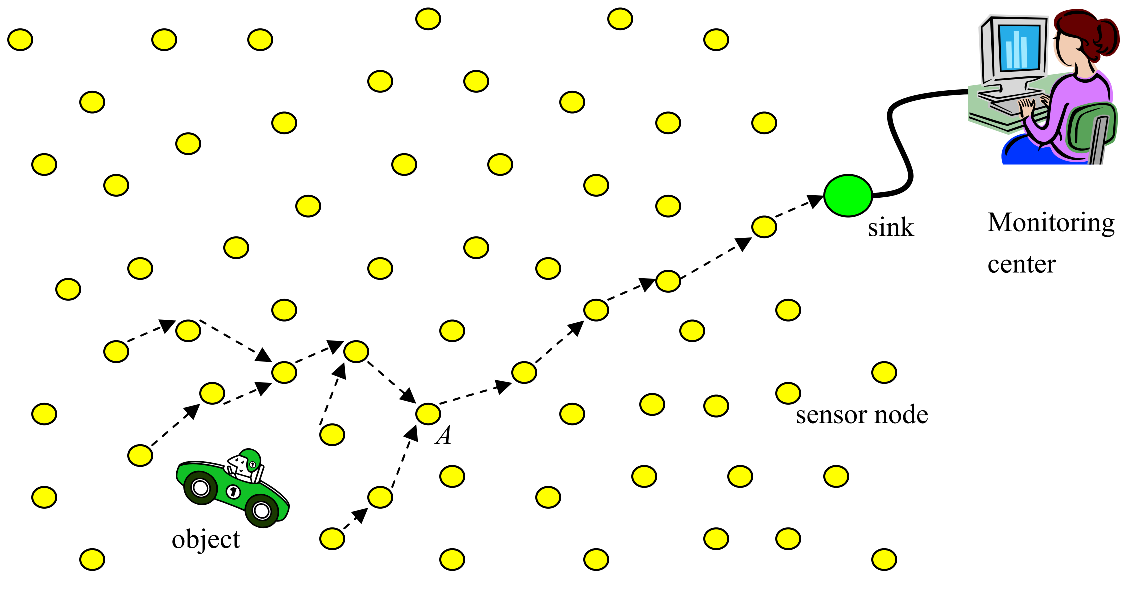

Figure 2-3. In-network processing of WSN

However, source nodes may be scattered among a WSN randomly. Some source nodes may be close to each other, and others may be far away from each other. It is clear that only those source nodes that are close to each other should be clustered together for in-network processing. If source nodes are far away from each other, in-network processing is of no benefit. Therefore, in-network processing should only run inside each local cluster. Nevertheless, how a WSN can be separated into several clusters is still a problem that needs to be solved.

Most schemes simply cluster sensor nodes by their *location*s, regardless of whether these nodes are source nodes or not. Some schemes [Ye02, Sha05] divide the network area (or the monitoring area) into square patches when the network is constructed, and each patch corresponds to a cluster. In [Sto05], a localization system is deployed over WSNs. In their design, a spotlight device (a localization-related hardware) is installed at an actuator. The spotlight device steers a laser light to measure the *location* of a node, and then informs the node of the measurement result. After that, sensor nodes can be clustered by their measured *location*s. The drawback of this clustering scheme is that it depends on a reference (a spotlight device) outside the WSN.

Training Protocol is an innovative design on node clustering [Wad05]. In a WSN, two sensor nodes are selected to initiate a training process. During the training process, each of those two nodes establishes the shortest routing path tree around itself. After the training process, every sensor node should maintain two numbers, which are hop counts to those two selected sensor nodes. In fact, these two numbers (hop counts) can be used as *location* coordinates of the node. The nodes with the “similar” coordinates are grouped together to form a cluster. Among these clustering designs, a cluster header should be selected inside each cluster. The cluster header can be elected or be predefined within the cluster. Inside a cluster, sensory data converge to the cluster header for in-network aggregation.

Some other designs cluster nodes by the routing paths towards the sink node, For example, the shortest routing path tree of the sink node [Kri02] is explored for the node clustering. According to this design, sensor nodes that locate at the same branch are clustered together. Along the delivery path, sensory data from source nodes that are far from the sink node may pass by some other source nodes near the sink node. As a result, sensory data can be aggregated along their way.

Also, node clustering can adapt to environmental conditions. There are some designs that cluster WSNs by the intensity of the sensing signal. In these designs, sensor nodes are clustered by the boundary of the physical incident that WSN is monitoring. In this way, node clusters are associated with the *location*s of physical incidents. The boundary of a node cluster is identified by the signal amplitude sensed by sensor nodes (i.e. physical signal intensity). If the amplitude of sensed signal is lower than the threshold, the node must be outside a cluster. Within a node cluster, the cluster header is still elected by the amplitude of sensed signal, and the node with the highest amplitude would win [Fan03]. A similar election mechanism is also described in GRAB [Ye02], in which the node of “center of signal” (CoS) is elected by the strength of sensed signal. Furthermore, only the sensory data from the node of CoS has been required to be delivered to the sink node.

2.2.3.2 WSN Routing

Most routing schemes over WSNs include in-network processing, in order to publish sensory data (or incidents) to the sink node efficiently [Kri02, Ye02, Sha04, Bea03a, Bea03b, Shi05, Sha05]. In [Kri02], three routing schemes, relative to the energy expense and the time delay, are proposed. The first routing scheme is named “Center at Nearest Source (CNS).” According to the scheme of CNS, a sensor node can be selected as the cluster header, only if the node is within the cluster and nearest to the sink. The second routing scheme is “Shortest Paths Tree (SPT).” According to the scheme, sensory data are delivered along the shortest routing path tree of the sink node, and can be aggregated on their way. The third one, “Greedy Incremental Tree (GIT),” is an iterative scheme that runs round by round. At each round, the source node that is nearest to the current routing tree is added through the shortest routing path. The routing tree grows round by round, until all source nodes are included.

In [Int03, Bis05], a query packet is diffused from the sink node, in order to probe the routing path tree of the sink node. Sensor nodes are activated by the query packet, and sensory data (or incidents) are delivered along the reverse path of the query diffusion. In addition, sensory data are aggregated along their delivery path. In [Sha04], the data sampling is triggered by the timer and by the variation of sensed signal. Similar ideas can also be found in [Bea03a] and [Bea03b]. In [Shi05], a virtual infrastructure, “Rail,” is constructed for in-network processing. “Rail” acts as a rendezvous area of sensory data, and all data are aggregated within the “Rail.”

As is the case with the designs of node clustering, the routing paths can also adapt to environment conditions. In [Hon04], each node forwards sensory data, based on the remaining energy of both itself and its neighbors. In [Gan04], packets of sensory data are agent-based. These agent-based packets determine their routing paths hop by hop. Whenever an agent-based packet arrives at a node, its agent section will instruct the node to calculate its next delivery stop, based on the remaining energy of the node and the transmission expense of the next hop.

Local target protocol (LTP) [Cha05] could help sensory data packets detour concave holes of WSNs. During the “search phase” of LTP, the node locates its neighbors that are closer to the sink, and the utmost neighbor is selected. During the “direct transmission phase” of LTP, the sensory data packet is sent to this utmost neighbor. If no neighbor is closer to the sink than the node itself during the “search phase,” and the sensory data packet has not arrived the sink node, the “backtrack phase” is invoked. During the phase, the data packet is sent back to the node where it came from.

Overall, the technology of in-network processing can greatly reduce the expense of sensory data publishing (or incident publishing). With WSNs clustered, in-network processing can be conducted. Currently, there are three types of schemes regarding node clustering. They are pre-defined clustering, routing path clustering and signal amplitude clustering. For the routing scheme, the issue of “routing within a cluster” is rarely discussed. In the dissertation, the author follows the schemes of signal amplitude clustering and proposes a routing scheme that can not only work within clusters for in-network processing, but also smoothly collaborate with existing global routing schemes.

Summary

Towards network forensic support, the discussions are separated into two parts, i.e. incident logging and incident publishing. For incident logging, *location* is one of the key components to accurately record happenings. Thus, research efforts related to the *location* and its application (i.e. node authentication) are inspected. Afterward, the issue of incident publishing is discussed under three different circumstances (i.e. general mobile networks, cellular networks and wireless sensor networks). Research efforts presented mainly focus on decreasing the communication cost of incident publishing.

Chapter 3 Localization and Authentication in Network Topology Spaces

In order to support network forensic, the log file and the logging mechanism must be researched firstly, because they are the bases of any forensic analysis. In this chapter, the research of incident logging is focused on the *location* and one of its applications (i.e. node authentication). The research is conducted over a typical model of mobile ad hoc networks. The objective of the research is to provide a solution to localize mobile nodes in a network topology space. Meanwhile, the solution of node localization is leveraged across the process of node authentication, aimed at defending the attack of node ID spoof.

3.1 Problem Description

In general, a Mobile Ad hoc Network (MANET) consists of a group of mobile nodes which can communicate with others via a wireless channel. To be an open system, a MANET requires that all nodes be able to join or leave the network freely. However, this is problematic in terms of MANET security because malicious nodes can join MANET freely and anonymously. Inside a MANET, the malicious nodes can jam wireless channels by flooding useless packets, or just shuffling/dropping passing packets that should be forwarded in an orderly fashion. Also, a malicious node may disguise its ID, and abuse services that are provided by the MANET without penalty.

In this chapter, a type of attack — node ID spoof is addressed, and the solution to this attack is presented. This solution differs from other strategies that rely on keys and key management systems. That is, every node joins a MANET with a unique ID number, as is the case with a person’s name that can be used to identify whom the person is. However, a malicious node may change its ID after a while to avoid the penalty of misbehaving. In addition, there is almost no expense for a node to change its ID number because of the ad hoc property that MANETs possess, even though this constitutes a certain abuse of the MANET’s freedom. By spoofing its identity, a malicious node may also apply services that are provided by other nodes and create chaos without a penalty. In this instance, a mechanism to make every node (and its behaviors) accountable, responsible, and traceable is presented. Under this mechanism, a process of node authentication has also been designed for service supplies.

3.2 High Level Solution Description

The mechanism requires that each node of a MANET one-hop broadcast its current background information (neighbor list) periodically. An example is useful to illustrate this mechanism. A person lives in a community, which is similar to a node in a MANET. The person is named Tom, just like a node with its ID number. Moreover, Tom also has background information, such as which high school he graduated from, whom his parents are, and where his home-town is located. Based on Tom’s background information, it is always possible to identify him, regardless of whether his name is still Tom or it is changed to Jerry.

For a mobile node of a MANET, its neighbors during a time period can be viewed as its background information at that time. To be recognized or identified by other nodes, the node should not only maintain its background information (neighbor list) during the time by itself, but should also distribute copies of its background (neighbor list) to other nodes. This is similar to a case in which someone states that he knows someone else due to the fact that he knows the person's name as well as other basic information of the person, such as the appearance, the friends and the place the person often to hang around. By broadcasting its background information (neighbor list), a node, which intends to spoof its identity later, also needs to spoof those distributed background copies. However, changing the background copies maintained by other nodes is somewhat complex. As a result, the expense of ID spoofing increases.

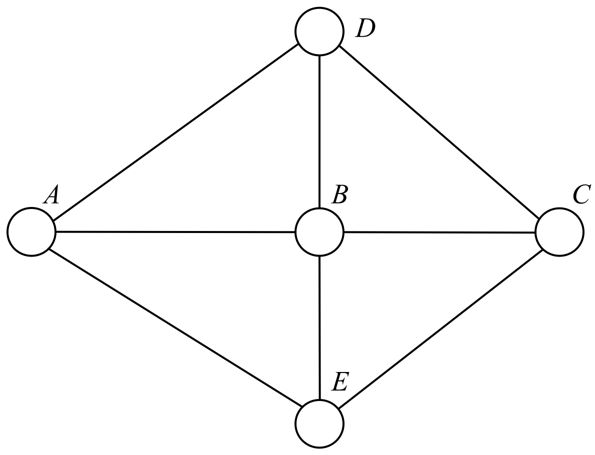

Figure 3-1. A special case when nodes are with same neighbors

There is a special case as shown in Figure 3-1. In this case, node *A* and node *C* are with same neighbors, and so, node *C* has a chance to impersonate node *A* by simply changing its own ID. Based on Figure 3-1, any person can easily identify the fact that node *A* and node *C* are with same neighbors.

However, this figure is not available to node C immediately. In fact, when node *C* is outside of the broadcasting range of node *A*, it cannot know the neighbor list of node *A*. In order to get information of neighbors surrounding node *A*, node *C* must move within the broadcasting area of node *A* and eavesdrop, and consequently cause the topography changed.

It is true that node *C* may still impersonate node *A*, or our design would still possibly be hacked. For example, node *C* may move into the broadcasting area of node *A*, and silently eavesdrop the background broadcasting from node *A*. After that, node *C* moves out of the broadcasting area of *A*, and to the position shown in Figure 3-1. In such a case, node *C* can mimic node *A* for a while with same background, until the relative position between node A and node B, D, E varies. This is a limitation of our design, and would be researched further in the future work.

In addition, node *C* may collaborate with node *B* (or *D*, or *E*), in order to get the neighbor list of node *A* and impersonate node *A*. During the collaboration, node *B* may receive the broadcasted neighbor list from node *A*, and informed node *C* secretly. Because of the limit of this research, these collaborations among malicious nodes would not be considered. Also, node *C* may obtain the neighbor list of node *A* through some special capability that other normal nodes don't possess. In this dissertation, we assume that most nodes are honest and reliable to follow our background broadcasting design, and all nodes are with same capability. In addition, malicious nodes cannot collaborate with each other.

Specifically, each node should broadcast its ID and its current neighbor list (a set of ID numbers) periodically. In the mean time, each node should maintain a background table, based on the received background packets. The background table of a node should include the node's self-background (including all neighboring IDs which are one-hop from the node), and its neighbor's background (which are one-hop from this neighbor). The node's background table should be logged periodically along with time stamps. Whenever the node needs a service, the service provider (another node) requests the node to submit its background (neighbor list) at a particular time point. Based on the node's response, the node's neighbors at that time will be referenced to check whether the node's response is correct or not. In this way, any intended ID spoofing can be identified.

The outline of the chapter is as follows: Section 3.3 presents the mechanism of the background construction and the background table maintenance; Section 3.4 describes the procedure of the node authentication based on background checks; Section 3.5 investigates the overhead incurred by the designed mechanism.

3.3 Constructing the Background of Nodes

3.3.1 The Model

This chapter follows a commonly accepted MANET model. That is, a group of mobile nodes construct a multi-hop mesh network via its wireless radio. Whenever a node joins MANET, it should select a unique number to be its node ID. This can be achieved by following a similar mechanism to that of the Global Unique Identifier (GUID). It is assumed that all selected node IDs are evenly distributed in their namespace. After selecting the node ID, the node should take part in a periodic process of background broadcasting, which will be discussed in sub-section 3.3.2. Before the node leaves the MANET, it does not need to announce its departure. In addition, it is assumed that the storage space of each node is enough for incidents logging, and all participating nodes are time synchronized.

3.3.2 Background Broadcasting

Each node should broadcast its node ID and its current background (neighbor list) periodically. The time interval of background broadcastings depends on the nodes' average mobile intensity, the node density, and traffic loads. The size of the background broadcasting packet corresponds to the node density of the MANET. For example, each broadcasting area covers 30 nodes. Since each GUID needs 16 bytes, the size of the MANET's background broadcasting packets may be around 512 bytes long. A background broadcasting packet should include sections to carry these node IDs and a time stamp. Within a background broadcasting packet, the node's ID, from which the packet is broadcasted, is called the index ID. In this instance, only the packet format of the application layer is taken into account. When a node receives a background broadcasting packet, the information carried by the packet should be integrated into the "background table" (sub-section 3.3.3) that is maintained by the node.

3.3.3 Background Table Maintenance

Each node should maintain a table, based on the arrived background broadcasting packets. In addition, the table at each node should be logged with a time stamp when a new background broadcasting packet generates at the node. At each node, the background table should be indexed by all its neighbors' node IDs, and each entry of the background table is a list of all neighbors of the index node. For example, a MANET consists of five mobile nodes, and its current topography is shown in Figure 3-3.

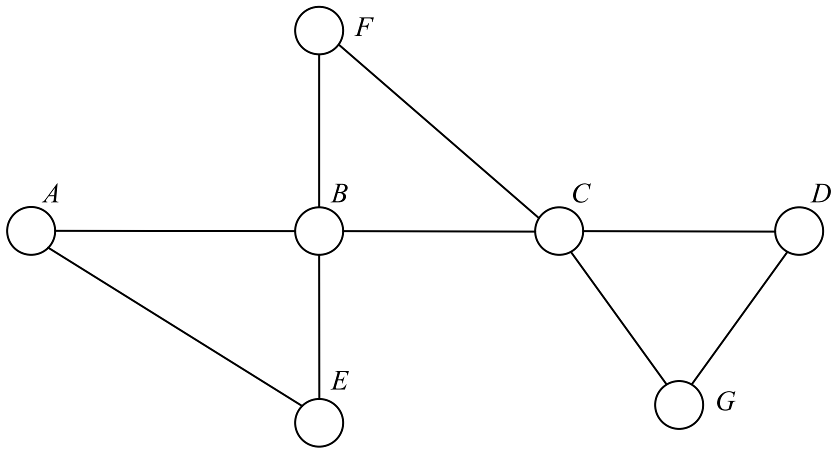

Figure 3-2. A MANET example for background table construction

The current background table at node *B* is shown in Table 3-1. The left-hand column of the table contains index nodes which are all neighbor node IDs of node *B*. The right-hand column of the table contains all node IDs of neighbors of each index node.

From topography shown in Figure 3-3, it is evident that node *A*, *C*, *E*, and *F* are neighbor nodes of node *B*, and they are all listed in the left-hand column of Table 3-1.

Table 3-1. Background table at node *B*

A	*B, E*
C	*B, D, F, G*
E	*B, A*
F	*B, C*

Node *A* has two neighbor nodes, node *B* and node *E*. Thus, for the row of node *A* in Table 3-1, nodes *B* and *E* are listed in the right cell. From Figure 3-3, nodes *B*, *D*, *F*, and *G* are neighbors of node *C*. Therefore, nodes *B*, *D*, *F*, and *G* are listed in the right-hand column corresponding to node *C*. Similarly, the other two cells that are indexed by nodes *E* and *F* can be filled in as well. A row of the background table should be updated when a background broadcasting packet is received. For example, node *B* receives a background broadcasting packet. The index node ID of this packet is node *E*, and the neighbor list carried by this packet includes nodes *B* and *G*. Node *B* should update its background table by filling *B* and *G* into the cell indexed by *E*. The updated table is shown in Table 3-2.

Table 3-2. Updated background table at node *B*

A	*B, E*
C	*B, D, F, G*
E	*B, G*
F	*B, C*

From the above example of the background table update, it is evident that a background broadcasting packet only updates one row of the table. In addition, each row of the table is associated with a time stamp, which may expire after two or three broadcasting intervals. The row should be deleted when it is expired. Whenever a background broadcasting packet is received, the corresponding row of the table should also have its associated time stamp updated by the time stamp of the packet. When a background broadcasting packet is received and its index node ID is new to the background table, a new row will be added to the background table. The index of this new row should be the index node ID of the received background broadcasting packet, and all neighbor node IDs carried by the packet should be filled in the right column for this new row.

3.3.4 Mathematics behind the Background Broadcasting

All nodes of a MANET construct the set $\boldsymbol{F}$={all nodes of the MANET}. For any node with its node ID "*x*," a topology space can be generated as $\boldsymbol{T_x} = \{ \mathbf{0}, \{x\}, \{\text{all } x\text{'s neighbors}, x\}, \boldsymbol{F} \}$. For the first three items of $\boldsymbol{T_x}$, **0** denotes the empty set; $\{x\}$ denotes the set that only node x is included in; $\{\text{all } x\text{'s neighbors}, x\}$ denotes the set that node x, together with all its neighbors are included in. Here, the node x is the core node of the topology space $\boldsymbol{T_x}$. In fact, this kind of topology space can be generated at each node of the MANET. For all of the topology spaces of different nodes, the first item "**0**" and the fourth item "$\boldsymbol{F}$" are the same. These topology spaces can differentiate from each other by the second item (for $\boldsymbol{T}_x$ is $\{x\}$) and the third item (for $\boldsymbol{T}_x$ is {all x's neighbors, x}). To measure the distance between each pair of the different topology spaces, we map these two items onto two vectors in the N-dimension space $\mathbf{R}^{\boldsymbol{N}}$. Here, N is the total number of nodes in the MANET. For the vector space $\mathbf{R}^{\boldsymbol{N}}$, each dimension corresponds to a node of the network. The coordinates of the vectors are coded by "1" or "0", in order to represent the corresponding node is or is not included.

For example, the topography of a network is shown in Figure 3-3. The topology space generated at node B: $\boldsymbol{T}_B = \{ \mathbf{0}, \{B\}, \{A, B, C, E\}, \{A, B, C, D, E\} \}$. Totally, there are five nodes in the network, and so a five-dimension vector space is used for the mapping. For this five-dimension vector space, let the first dimension corresponds to node A; the second dimension corresponds to node B; the third dimension corresponds to node C; the fourth dimension corresponds to node D; and the fifth dimension corresponds to node E. Therefore, the second item of $\boldsymbol{T}_B$, $\{B\}$, is mapped onto the vector $[\ 0\ 1\ 0\ 0\ 0\ ]^T$, as only node B is included. Similarly, the third item of $\boldsymbol{T}_B$, $\{A, B, C, E\}$, is mapped onto the vector $[\ 1\ 1\ 1\ 0\ 1]^T$, as in this case, only node D is not included.

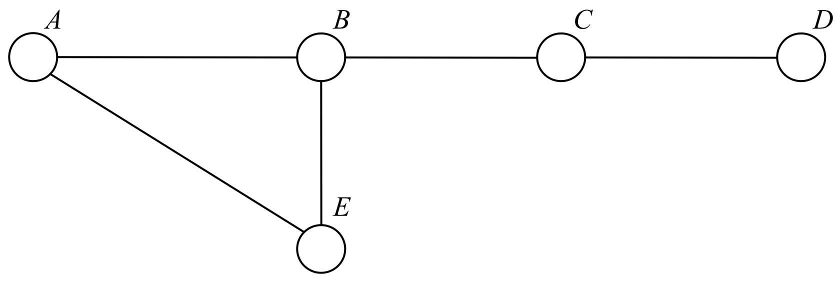

Figure 3-3. An example of network topography

Before the mapping, all nodes are sorted by their IDs and correspond to entries of the vectors from top to bottom (or from left to right if the vector is transposed). In this example, the first entry of the vector corresponds to node *A*, the second entry of the vector corresponds to node *B*, and so on. In addition, the first vector corresponds to the second item "{*B*}" of the space $\boldsymbol{T}_B$, and the second vector corresponds to the third item "{*A*, *B*, *C*, *E*}" of the space $\boldsymbol{T}_B$. If the node ID is included in the item of the topology space, the corresponding entry of the vector is set to be 1. Otherwise, the corresponding entry of the vector is set to be 0.

Following the example above, the second item of $\boldsymbol{T}_B$ is {*B*}, which is mapped onto the five-dimension vector $[\ 0\ 1\ 0\ 0\ 0\ ]^T$. The argument behind this mapping is that only node *B* is included in the item of $\boldsymbol{T}_B$, and only the second entry of the vector is set to be 1. The third item of $\boldsymbol{T}_B$ is {*A*, *B*, *C*, *E*}, which is mapped to the five-dimension vector $[\ 1\ 1\ 1\ 0\ 1]^T$. The argument behind this mapping is that only node *D* is not included in the item of $\boldsymbol{T}_B$. So, only the fourth entry of the vector is set to be 0, and all the rest entries are set to be 1. In this way, the space $\boldsymbol{T}_B$ is represented by these two vectors.

The benefit of using vectors to represent a topology space is that the distance between different topology spaces can be measured by the inner product of vectors. Moreover, the distance between any two topology spaces may represent the distance of the core nodes of those two spaces in a network circumstance. As a result, these two vectors can be used for localizing core nodes in a network topology space; even through the concept of the *distance metric* has not been introduced in the network topology space.

3.3.5 Security Issues on Background Broadcasting

Malicious nodes can join MANETs any time, and it cannot be assumed that those malicious nodes follow the proposed mechanisms. However, a malicious node can only affect the node's neighbors, and can do nothing about a node which is far from the malicious node. Overall, most nodes are normal in a MANET, and the percentage of malicious nodes should be less than 1. Nevertheless, the proposed mechanism of background broadcasting can still be hacked; however, the influence of this hack is confined to the local area.

3.4 Authentication

3.4.1 Background Authentication

Since a MANET is an open system, services given over a MANET should be accessed by any honest MANET member (node). For example, if there are two nodes, *X* and *Y*, node *X* is going to access a service provided by node *Y*. In this case, node *X* should send an application message to node *Y* first. After that, node *Y* requests node *X* to submit its neighbor list at a particular time point (say *t*). When node *X* receives the request, node *X* should extract its neighbor list at time *t* from its log files. This neighbor list includes all node IDs in the left-hand column of node *X*'s "background table" at time *t*. If node *X* just joins the MANET and has no history log at time *t*, the service requests from node *X* should be declined. If the submitted neighbor list only includes four or fewer node IDs, node *Y* also needs to decline the service requests from node *X*.

Whenever node *Y* gets the neighbor list of node *X* at time *t*, a background check starts. Node *Y* selects 3~10 node IDs from the submitted neighbor list as node *X*'s references, and sends the neighbor list, together with *X*'s node ID and time *t*, to each of node *X*'s references. Each reference of node *X* only needs to reply "correct" or "not correct," by checking the received list with the log file of itself. If more than 80% (or 90%) of the replies from node *X*'s references are "correct," node *Y* should approve node *X*'s service application. Otherwise, node *Y* should decline node *X*'s service application. Here, all numbers in the above descriptions, such as 3~10 node IDs and 80% (or 90%), can be adjusted and optimized in practice.

3.4.2 Mathematics behind Background Authentication

The neighbor list from node *X* is used to construct the two vectors of the topology space $\boldsymbol{T}_X$. At a reference node of node *X*, the 1st and 2nd vector of $\boldsymbol{T}_X$ (named R_{1X} and R_{2X}, respectively) can be generated from the received neighbor list of node *X*. After that, the reference node uses R_{1X} to search for the logged neighbor list of node *X*. Based on the logged neighbor list of node *X*, the reference node constructs the 2nd vector of $\boldsymbol{T}_X$ again (named L_{2X}). Normalized inner product between the vector R_{2X} and L_{2X} is calculated and compared with a threshold to identify whether the received neighbor list is correct or not.

For example, the Figure 3-3 shows a MANET. Node *B* has its background checked. By the neighbor list from node *B*, we have R_{2B} = $[1\ 1\ 1\ 0\ 1]^T$. Assuming that at the reference node *A*, L_{2B} = $[1\ 1\ 0\ 0\ 1]^T$, the normalized inner product is

$$\frac{\langle R_{2B}, L_{2B} \rangle}{\|L_{2B}\| \cdot \|R_{2B}\|} = \sqrt{3}/2 = 0.866. \qquad \text{(3-E1)}$$

If the given threshold is set to be 0.85, the node A should reply "correct." The difference between R_{2B} and L_{2B} may be caused by the nodes' movements and/or noise. In this example, the threshold is set to be 0.85 arbitrarily. In practice, the threshold can be generated by statistics, based on different moving patterns of nodes and noise.

3.4.3 Recursive Background Authentication

In fact, the reference nodes can be questioned as well. After a reference node replies the background check of node *X*, node *Y* can even ask this reference node to submit its neighbor list at another particular time point t_2. In this way, "Background authentication" can be incurred recursively. As a result, the possibility of multiple malicious nodes cooperating with each other to cheat can be repressed.

3.5 Simulation

In this section, the effects of background broadcasting on the communication traffic are investigated, coupling with the change of node density and mobile intensity. The effects are measured by the delivery delay and the packet delivery ratio. In the simulation, it is assumed that the interval of background broadcastings is three seconds. This time interval of background broadcasting comes from the fact that mostly, five meter per second is the upper bound of node's moving speed, and the transmission radius of the broadcasting is about two hundred fifty meters.

3.5.1 Simulation Model

Simulations are conducted on an NS-2 simulator. The bandwidth of wireless channels is set to be 11Mb/sec; the broadcast radius of each node is set to be 250m. The network area of all simulations is 1000m × 1000m. Ad Hoc On-demand Distance Vector (AODV) is selected as the routing protocol. Two applications, CBR (Constant Bit-Rate) and FTP (File Transport Protocol), are used to mimic traffic loads over the MANET. The packet size of CBR is fixed at 512 bytes.

Simulations run under two circumstances: static circumstance (all nodes are static at their positions), and dynamic circumstance (all nodes move randomly inside the network area). For the static circumstance, the traffic load varies from 250Kb/sec to 10Kb/sec, and the hop count from the source node to the termination node varies from 1 to 10. The node density of the static circumstance is set to be n = 50, 30, and 20 nodes per broadcasting area. The node density of the dynamic circumstance is set to be n = 30 nodes per broadcasting area.

3.5.2 Simulation Results

As shown in Figures 3-4, 3-5, and 3-6, when n = 50, the mechanism of background broadcasting affects network performance significantly since the packet delivery ratio without background broadcasting is higher than that with background broadcasting.

However, in most implementations, MANETs are constructed with their node density *n* = 20 or 30 nodes per broadcasting area. When *n* = 20 or 30, the effect of the background broadcasting on packet delivery ratio is limited. Especially when the traffic load is less than 50kb/s, the packet delivery ratio of CBR with background broadcasting is almost the same as that without background broadcasting. Therefore, it can be concluded that in practice, the mechanism of background broadcasting will not impair MANET performance.

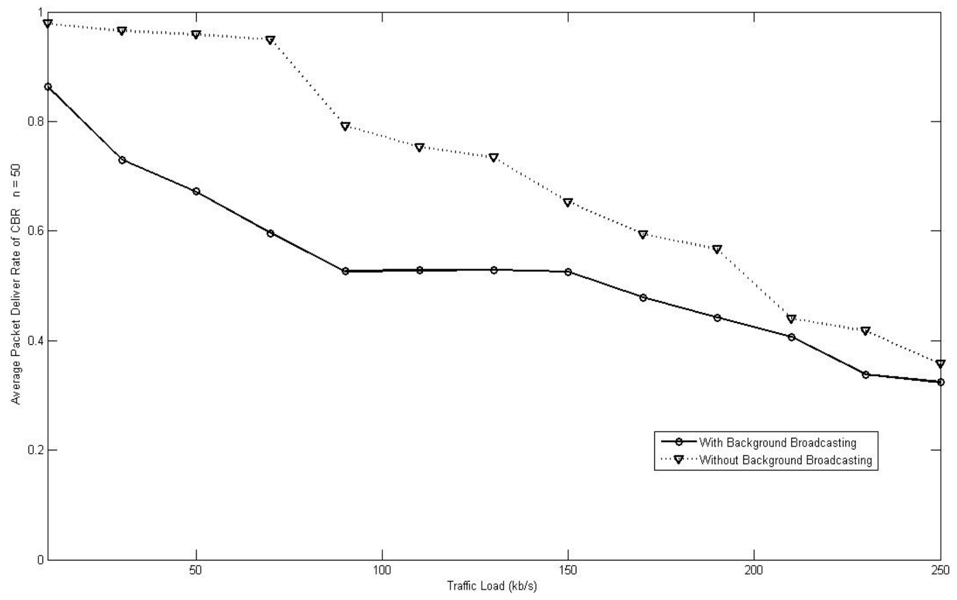

Figure 3-4. Effects of BB on packet delivery rate when *n* =50

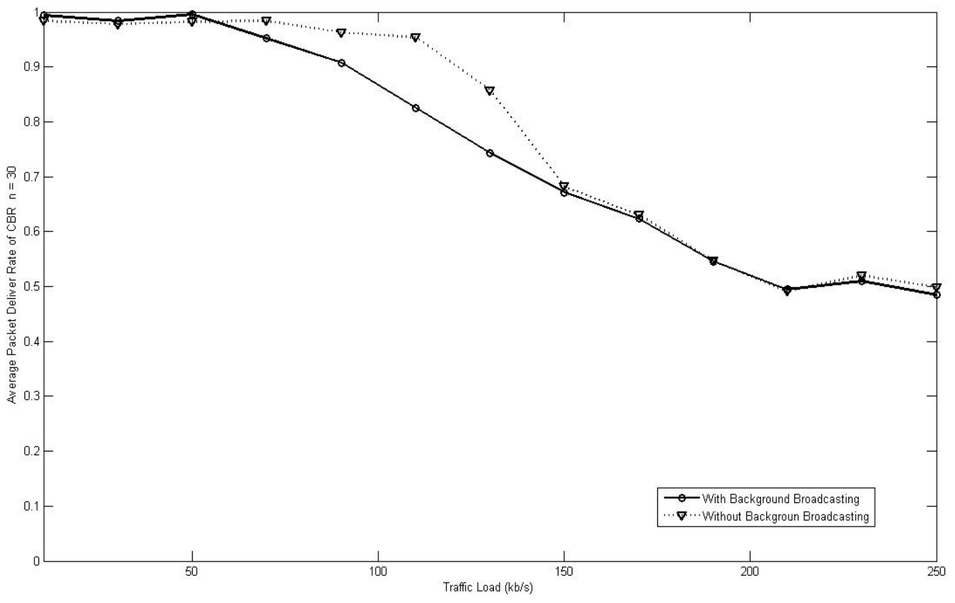

Figure 3-5. Effects of BB on packet delivery rate when *n* = 30

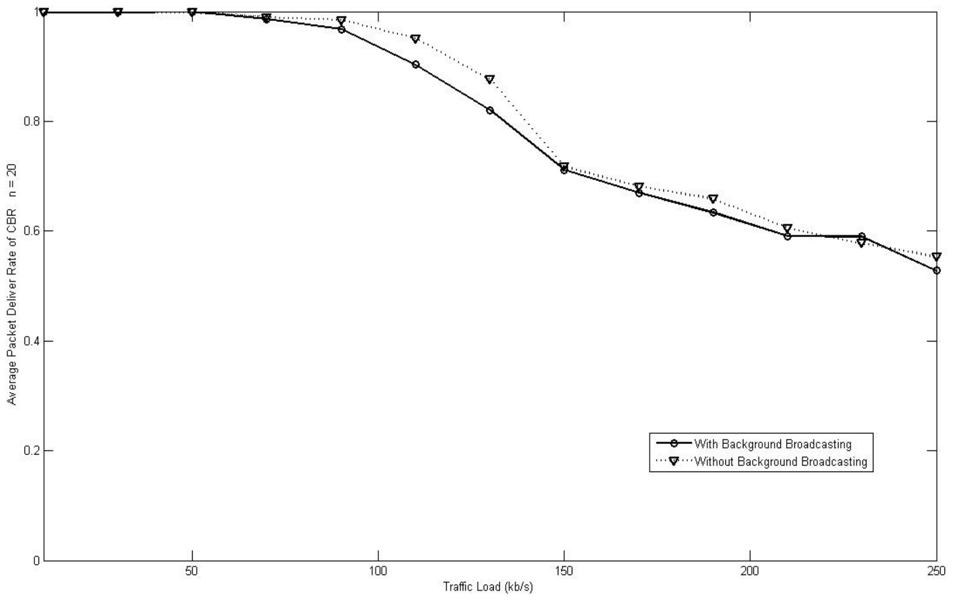

Figure 3-6. Effects of BB on packet delivery rate when *n* = 20

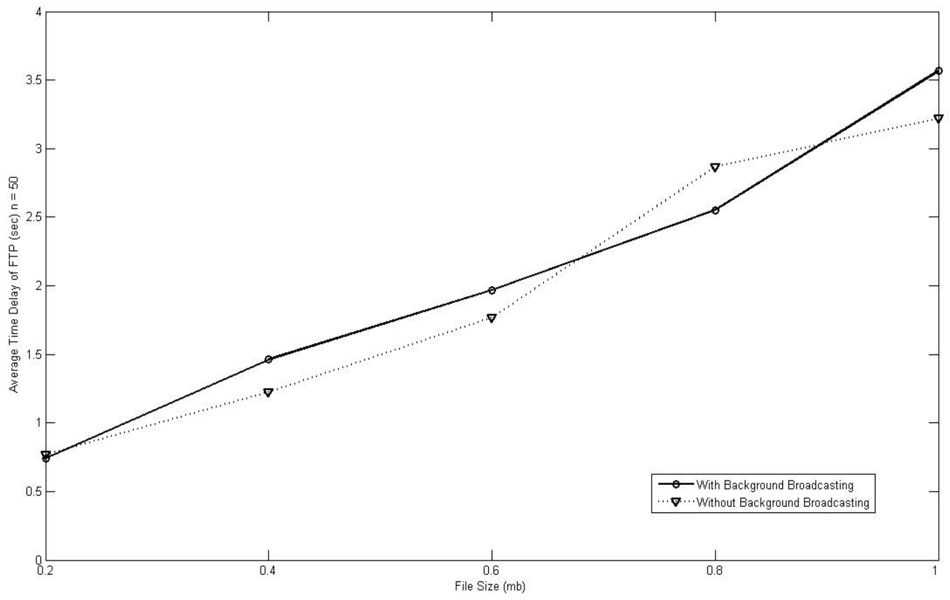

Figure 3-7. Effects of BB on delivery delay when *n* = 50

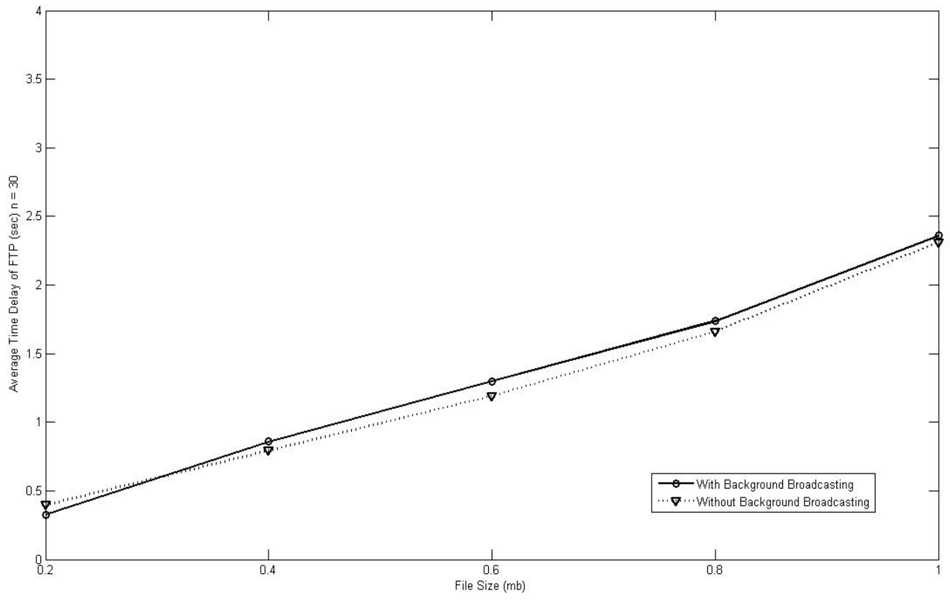

Figure 3-8. Effects of BB on delivery delay when *n* = 30

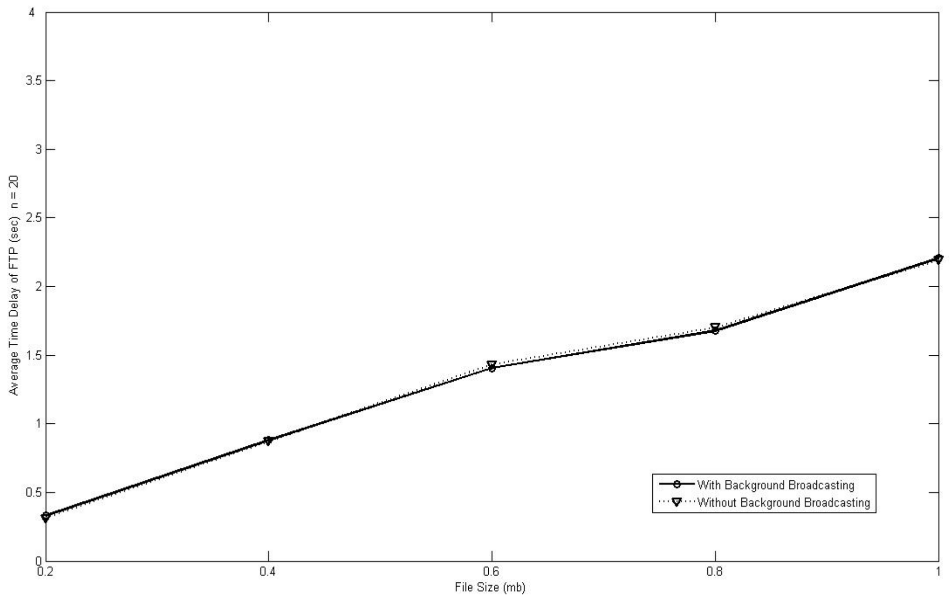

Figure 3-9. Effects of BB on delivery delay when n = 20

As shown in Figures 3-7, 3-8, and 3-9, when n = 50, the background broadcasting may delay packet delivery somewhat for FTP applications. However, in most practical implementations in which MANETs are constructed with their node density at n = 20 or 30 nodes per broadcasting area, the packet delivery delay with background broadcasting is almost the same as that without background broadcasting. Therefore, the investigation of FTPs' time delays also supports the conclusion that the mechanism of background broadcasting will not impair MANET performance.

For the dynamic circumstance, the speed of a node is < 5m/sec, and the hop count from the source to the destination is 2. From the Figures 3-10 and 3-11, background broadcasting would not impair the performance when the traffic load is less than 300kb/s.

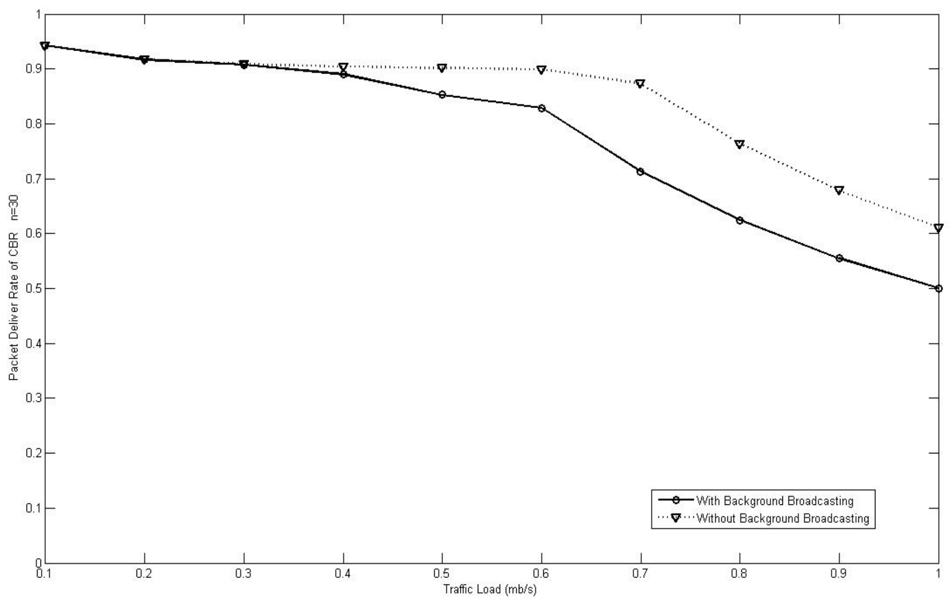

Figure 3-10. Effects of BB on delivery rate when nodes are moving

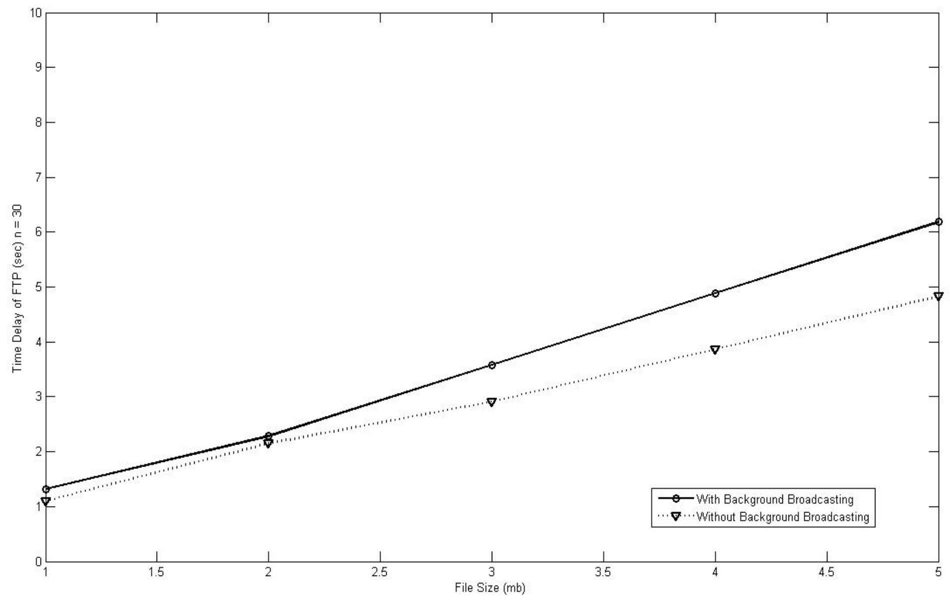

Figure 3-11. Effects of BB on delivery delay when nodes are moving

One thing need to be noted is that there is only 2 hops from the source node to the destination node. This is the reason that the mobile networks can take on such heavy traffic loads with such high packet delivery ratios, as shown in Figure 3-10.

Summary

For incident logging, the mechanism of node localization is discussed in network topology spaces. During the discussion, the concept of Background is introduced. The Background can localize mobile nodes without a *distance metric*. In the chapter, the Background is also applied to the node authentication. Because logs are the starting point of forensic analyses, incident logging is studied in this chapter. In the following chapters, the research would focus on the next stage of forensic support, i.e. the log publishing.

Chapter 4 Log Publishing over IP-based Mobile Networks

From now on, the log publishing (or incident publishing) is taken into account. In this chapter, the research is conducted over a general model of mobile networks. The objective of the research is to provide a solution to organize log files into a distributed database. Meanwhile, a mechanism is designed to limit the communication cost of the process of log publishing.

4.1 Problem Description

As discussed in Chapter 1, a network incident can be viewed as an aggregation of several raw incidents. In addition, these raw incidents or their aggregations should be logged for later forensic analyses. In the incident that a hacker compromises a remote computer via internet, one can trace back this incident by collecting all related logs and analyzing these collected logs. Further, the traffic loads over a particular link vary from time to time, and the only way to investigate the traffic variation of this link is to collect logs along the link and analyze them. Thus, log files are the only materials (documents) that can be worked on to conduct network forensics.

In addition, all logs distributed throughout the entire network need to be organized. For example, there is a raw incident in which a packet from a source node to a termination node is delivered. This raw incident is logged at the source node, the destination node, and all intermediate relay nodes separately. To reconstruct the delivery path of the incident, the logs from those different nodes should be collected (the source node, the destination node, and all intermediate relay nodes). However, if all logs related to the incident are not organized, but are scattered randomly throughout the entire network, the process of collecting those logs related to the incident will be very difficult. Therefore, it is necessary to have all distributed log files organized in order to facilitate network forensics.

Many designs have been proposed in the research field of network forensic support. The typical two designs are PPM (probabilistic packet marking) and SPIE (Source Path Isolation Engine), which emphasize applications over fixed networks. According to the design of PPM, each packet will be marked probabilistically at each node along its delivery path. In fact, the marks inserted into the packet can also be treated as logs. Specifically, the logs (marks) travel along the packet delivery and converge to the termination node of the packet. In this way, related logs (marks) generated at different nodes can easily be organized at the termination node.

Nevertheless, SPIE suggests that passing packets should be logged at local nodes. To reduce the volume of logs, SPIE uses blooming filters to hash the content of packets. In contrast to organizing logs in advance, SPIE leverages the static topography of fixed networks to collect log files related to a particular delivery incident. According to the design of SPIE, the termination node of the packet should broadcast first to probe the neighbor nodes that possess related logs. This probation would be conducted hop by hop until the source node is reached.

Unfortunately, both designs of SPIE and PPM can not be deployed over MANETs directly. SPIE is obviously not suited for MANET forensics because of the dynamic topography of MANETs. Inside a MANET, the nodes involved in a packet delivery may move anywhere afterwards. To trace the packet's delivery path, the probing mechanism of broadcasting hop by hop does not work. In fact, log files of a particular incident (packet delivery) will scatter throughout the whole MANET area without any fixed pattern. Just as the typical intrusion illustrated by Figure 1-2 and in the section 1.1, the public server and all relay routers may move everywhere under a mobile circumstance, and one-hop broadcasting may not reach and probe the previous step of the attack. Furthermore, any attempt to broadcast the probation throughout the network cannot work well because of the huge cost.

Even for PPM design, which seems much better for a mobile environment, there still exists a formidable problem. Specifically, the destination nodes of malicious intrusions are most likely to be victim nodes and most likely to be compromised or tampered with. Consequently, those intrusion-related marks (or logs) that converged and recorded at the victim nodes would not be safe, since they could be easily destroyed or simply erased after the successful attacks. As a result, forensic analyses have no way to be conducted with all these intrusion-related logs destroyed.

Logically, our design focuses on supporting forensic analysis over mobile networks with dynamic topology. Moreover, our design would avoid driving all intrusion-related logs to the destination nodes which are potential victim nodes of malicious intrusions. Particularly, our design is fostered from the most popular one of P2P technologies, i.e. Distributed Hash Table, and customized to suit the needs of sorting or organizing distributed logs.

DHT (Distributed Hash Table) [May02, Rat01, Row01, Sto01, Zha03] technology offers an immediate solution to organizing the distributed log files over MANETs. However, problems still exist. All the log files maintained at different nodes are a kind of resource, and can be published based on a DHT structure, such as CAN (Content Addressable Network), Chord, Pastry or Tapestry, etc. Based on the DHT structure, logs related to a particular incident can be easily found and collected from different nodes.

A problem occurs when an attempt is made to publish these log files, since a publishing process is also a network incident and also needs to be logged and published. As a result, a publishing process may incur recursively without an end. In addition, we cannot fix this recursive problem by only publishing "regular incidents" and ignoring "publishing incidents." If nodes only log and publish regular packets, but ignore publishing packets to avoid recursive publishing, hackers can also launch a flood attack by solely using publishing packets. Consequently, this flood attack can not be traced back because it has not been logged.

In this chapter, an innovative scheme has been proposed to deal with the problem of organizing logs. First, all log files distributed at different nodes must be organized under a DHT structure. In order to avoid recursive publishing, the proposed design links each packet's delivery probability with its publishing probability. Whenever a packet is generated at a node, a value between (0, 1] should be assigned by the node. This assigned value, deemed delivery rate, is immutable for the lifetime of the packet.

The packet will carry its delivery rate along its delivery path. At each passing node, a packet will be forwarded with the probability of its delivery rate, or will be dropped with the complementary probability of its delivery rate. In addition, whenever a packet arrives at a node, a publishing packet should be generated at the node to publish the incident of the packet arrival. This publishing packet also needs to be assigned a delivery rate, which must be less than the delivery rate of the arrived packet. In this way, recursive publishing processes may still exist, but will eventually die out.

The outline of the chapter is as follows: Section 4.2 models logging and publishing processes under the typical DHT designs; Section 4.3 presents the designed scheme, analysis results, and optimization based on the modeling; Section 4.4 investigates the proposed scheme by simulations.

4.2 Theoretical Analysis:

4.2.1 Modeling

A commonly accepted MANET model is followed in this instance. A MANET consists of numerous mobile stations, and each mobile station is simply called a "node." Communications among nodes can be carried out via wireless radio, and radios equipped at nodes operate with the same power. Thereafter, two nodes, each of which locates outside the other's radio coverage, may communicate with each other by transmission relay nodes, locating between the two nodes. The whole MANET is assumed to be connected. This implies that there is always a routing path between any two nodes of the MANET. Each node has full computing capability, which suggests that each node may play a role as a server, a client or a router.

At each node, only incidents of packet arrivals are taken into account. As previously discussed, any network incident is an aggregation of several raw incidents. Generally, a raw incident refers to a packet incident that is transmitted from the source node, through intermediate relay nodes, to the termination node. At the node level, a raw incident can be described as the following: at a particular time and place, a packet is/is not forwarded or received at a particular node. In this Chapter, only the raw incident, "a packet arrived at a node," is considered. In fact, it is trivial to extend analysis to other types of raw incidents, such as "a packet is delivered at a node," "a packet is not arrived at a node," etc..

Assumption 1: *At each node, any raw incident should be logged, and the storage capacity of the node is unlimited.* (4-A_1)

As the previous discussions suggest, only raw incidents of packet arrivals are considered. Therefore, all logged incidents at a given node should be given the following format: "at a particular time and place, a particular packet arrives at the node." In addition, the effect of the storage capacity of nodes is omitted. This is used for incident logging because this chapter focuses on the process of log publishing first and foremost, (instead of the process of “incident logging.”) In addition, that the price of storage media has dropped substantially in recent years is another factor that was taken into account.

Assumption 2: *Any logged incident at a node should be published, regardless of whether the incident is incurred by a regular packet arrival or a publishing packet arrival.* (4-A_2)

Based on the typical DHT designs, an overlay structure is built for logs publishing. Each logged incident should be hashed first. After that, the publishing packet for publishing this incident should at least carry the hash result of the incident, the node ID, and the timestamp. Under the overlay structure of the DHT, the hash result determines the termination node where the publishing packet should be sent. Each logged incident includes four items: time, place, node ID and the content of the received packet. Only the item of the packet content should be hashed. In this way, the logged incidents with same packet content will be published at the same node under DHT architecture. In addition, a publishing packet must generate at the node whenever a raw incident occurs. Moreover, the publishing process is a one-direction packet delivery (UDP application), and does not guarantee whether the publishing packet will reach its termination node or not.

Assumption 3: *For any packet delivery, the hop count from the source node to the termination node is fixed to be m.* (4-A_3)

Clearly, the hop count of each packet delivery varies from case to case. However, since a MANET always has limited coverage area, the largest hop count of the MANET is finite. Thus, the hop count of every packer delivery is also a finite number with less than or equal to the largest hop count. To clarify the analysis, *m* denotes this finite number of every packet delivery.

4.2.2 Packet-Defined Probabilistic Delivery (PDPD or PD2)

Rule 1: *Each packet should be assigned a value to indicate the delivery rate of the packet.* $(4\text{-}R_1)$

The value, called "delivery rate," should be a positive number less than 1. The delivery rate of a packet would not change during the lifetime of the packet. In order to guarantee the backward compliance of the PD2 scheme, the default delivery rate is set to 1 for packets without a "delivery rate."

Rule 2: *At each node, a packet will be forwarded (or delivered) with a probability equal to the delivery rate of the packet.* $(4\text{-}R_2)$

When the delivery rate is 1, the packet should definitely be forwarded. If the delivery rate equals 0.5, there is an equal chance that the packet will be forwarded or dropped. If the delivery rate is 0, the packet should be dropped immediately. Even at the source node of a packet, the Rule 2 still applies. According to Rule 2, when the delivery rate is less than 1, the probability that the packet reaches its termination will decrease, as long as the hop count (from the source node to the termination node) increases. For example, the delivery rate of a packet is 0.8. When the hop count from the source node to the termination node is 3, the probability that the packet can reach its termination node will be 0.8^3. When the hop count of the packet reaches 5, the probability of the packet's successful delivery will be 0.8^5, which is less than 0.8^3.

Rule 3: *A function $f(\cdot)$ should be shared by all nodes to calculate the delivery rate of publishing packets.* (4-R_3)

A packet arrival is an incident, and should be logged at the node. Subsequently, a publishing packet generates and corresponds to this arrival incident. Similarly, this publishing packet should have its own delivery rate. The scheme of PD2 requires that the delivery rate of the publishing packet should be associated with the delivery rate of the received packet by the function $f(\cdot)$. For example, if the delivery rate of the received packet is p, the delivery rate of the publishing packet is equal to $f(p)$. In addition, $0 \leq f(p) < 1$. Detailed discussions about $f(p)$ will be given in the subsection 4.3.1. Regardless of whether the received packet is a regular one or just another publishing packet, the scheme of PD2 will treat it in the same way.

4.3 Analysis

In this section, we introduce a raw incident of a packet delivery. The packet is going to be delivered from the source node to the termination node after m hops (4-A_3). In addition, the packet has its delivery rate at p (4-R_1). Recursive publishing processes would be incurred by this packet delivery, and these recursive processes are split into a series of recursive rounds. A recursive round means that all publishing packets generated at the same recursive round should have the same delivery rate.

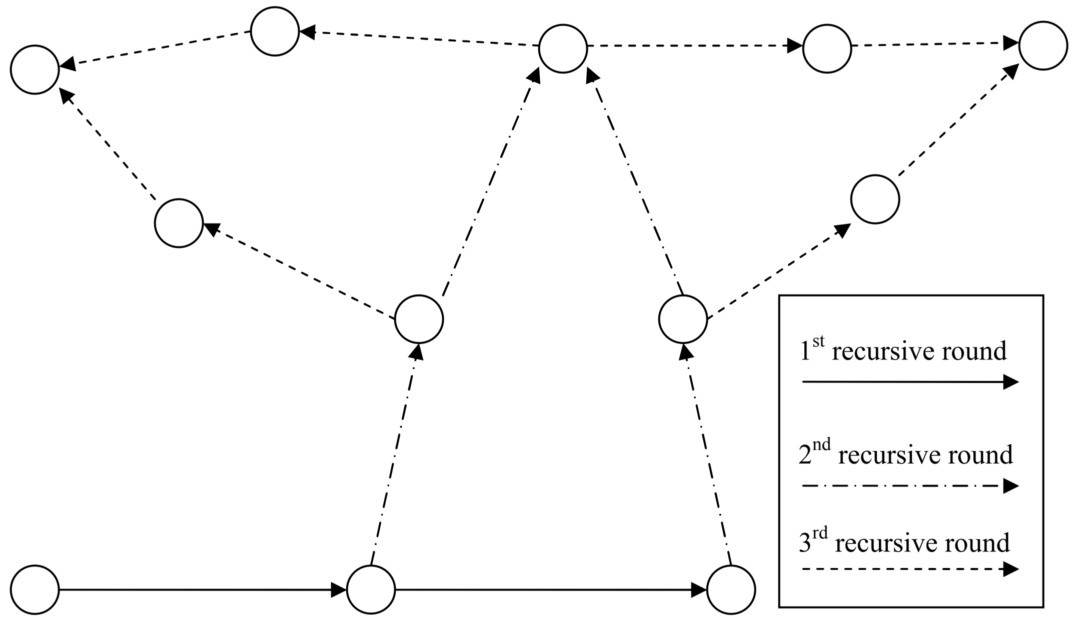

Figure 4-1. The first three rounds of the recursive publishing process

For example, a packet is delivered with the delivery rate p. The process of this packet delivery belongs to the first recursive round, because all publishing packets corresponding to this packet delivery share the same delivery rate $f(p)$. Next, the publishing packets generated at the first recursive round will incur new publishing packets. This process belongs to the second recursive round. In addition, the publishing packets of the second recursive round have the delivery rate $f(f(p))$ or $f^{(2)}(p)$, according to the rule (4-R_3). The first three recursive rounds of a packet delivery are illustrated in Figure 4-1.

4.3.1 The Total Number of Publishing Packets

First, it is necessary to calculate the average number of publishing packets incurred by a raw incident. As shown in Figure 4-1, the recursive processes are split into recursive rounds. The average number of publishing packets will be calculated round by round.

The first recursive round

By Rule 2 (4-R_2), the probability that the packet will be forwarded at least one-hop is p, and the probability that the packet will be forwarded exactly (only) one-hop is $P(1, 1) = p(1 - p)$. For $P(1, 1)$, the first "1" denotes the first round, and the second "1" denotes one-hop. Similarly, the probability that the packet will be forwarded exactly (and only) two-hops is $P(1, 2) = p^2(1 - p)$, and so on. The probability that the packet will be forwarded exactly (and only) $(m - 1)$ hops is $P(1, m - 1) = p^{m-1}(1 - p)$, and the probability that the packet will reach its termination node is $P(1, m) = p^m$.

Based on the assumptions of (4-A_1) and (4-A_2), a publishing packet will generate at a node whenever a packet arrives at the node. Therefore, the average number of publishing packets of the first recursive round N_1 is:

$$\begin{aligned} N_1 &= 1P(1, 1) + 2P(1, 2) + \ldots + (m - 1)P(1, m - 1) + mP(1, m) \\ &= \sum_{i=1}^{m-1} ip^i(1 - p) + mp^m \\ &= p(1 - p^m)/(1 - p) \qquad \text{(4-}E_1\text{)} \end{aligned}$$

In addition, according to Rule 3 (i.e. 4-R_3), all publishing packets generated at the first round should have the same delivery rate $f(p)$.

The second recursive round

For each publishing packet generated at the first recursive round, the hop count from its source node to its termination node is still m (4-A$_3$), and its delivery rate is $f(p)$. Similarly, for each publishing packet of the first recursive round, the probability that it will move exactly one-hop is $P(2, 1) = f(p)[1 - f(p)]$; the probability that it will move exactly two-hops is $P(2, 2) = [f(p)]^2[1 - f(p)]$. Also, $P(2, m - 1) = [f(p)]^{(m-1)}[1 - f(p)]$, and $P(2, m) = [f(p)]^m$.

On average, each publishing packet of the first recursive round may incur $f(p)\{1 - [f(p)]^m\}/[1 - f(p)]$ publishing packets at the second recursive round. Therefore, the average number of total publishing packets generated at the second recursive round N_2 is:

$$N_2 = N_1 f(p)\{1 - [f(p)]^m\}/[1 - f(p)] \qquad \text{(4-E}_2\text{)}$$

The third recursive round

For each publishing packet generated at the second recursive round, the hop count from its source node to its termination node is still m (A$_3$), and its delivery rate is $f^{(2)}(p) = f(f(p))$. Still, for each publishing packet of the second recursive round, the probability that it will move only one-hop is $P(3, 1) = f^{(2)}(p)[1 - f^{(2)}(p)]$. Similarly,

$P(3, 2) = [f^{(2)}(p)]^2[1 - f^{(2)}(p)]$,

… …,

$P(3, m - 1) = [f^{(2)}(p)]^{(m-1)}[1 - f^{(2)}(p)]$,

$P(3, m) = [f^{(2)}(p)]^m$.

On the average sense, each publishing packet of the second recursive round would incur $f^{(2)}(p)\{1 - [f^{(2)}(p)]^m\}/[1 - f^{(2)}(p)]$ publishing packets at the third recursive round. The average number of total publishing packets of the third recursive round N_3 is:

$$N_3 = N_2 f^{(2)}(p)\{1 - [f^{(2)}(p)]^m\}/[1 - f^{(2)}(p)]. \qquad \text{(4-E}_3\text{)}$$

The n-th recursive round

For each publishing packet generated at the $(n - 1)$th recursive round, the hop count from its source node to its termination node is m (A3), and its delivery rate is $f^{(n-1)}(p) = f(\ldots f(f(p))\ldots)$. Still, for each publishing packet of the $(n - 1)$th recursive round, the probability that it will move exactly one-hop is

$P(n, 1) = f^{(n-1)}(p)[1 - f^{(n-1)}(p)]$.

Similarly,

$P(n, 2) = [f^{(n-1)}(p)]^2[1 - f^{(n-1)}(p)]$,

… …,

$P(n, m - 1) = [f^{(n-1)}(p)]^{(m-1)}[1 - f^{(n-1)}(p)]$,

$P(n, m) = [f^{(n-1)}(p)]^m$.

On the average sense, each publishing packet of the $(n - 1)$th recursive round may incur $f^{(n-1)}(p)\{1 - [f^{(n-1)}(p)]^m\}/[1 - f^{(n-1)}(p)]$ publishing packets at the nth recursive round. The average number of total publishing packets of the n-th recursive round is:

$$N_n = N_{n-1} f^{(n-1)}(p)\{1 - [f^{(n-1)}(p)]^m\}/[1 - f^{(n-1)}(p)]. \qquad \text{(4-E}_n\text{)}$$

On average, the total number of publishing packets incurred by the incident of a packet delivery should be $N = N_1 + N_2 + N_3 + \ldots \ldots + N_n + \ldots \ldots$

For these recursive processes, the average number of total publishing packets N is expected to be limited. When $f(p) > p$, for any $p \in [0, 1]$, the average number of publishing packets of the second recursive round is greater than that of the first recursive round, and the average number of publishing packets of the third recursive round is greater than that of the second recursive round, and so on. In such cases, it is unavoidable for the number of publishing packets to explode. Therefore, it is required that $p > f(p)$ for any $p \in [0, 1]$. In fact, this requirement infers that $p > f(p) > f^{(2)}(p) > \ldots \ldots > f^{(n-1)}(p) > \ldots \ldots$, for any $p \in [0, 1]$. However, in order to guarantee that N be a limited number, this requirement ($p > f(p)$ for any $p \in [0, 1]$) is still not enough, and more conditions need to be fulfilled.

Theorem 1: For any $p \in [0, 1]$, and a limited number k, if $p > f(p)$ and $N_k < 1$, then N is limited, and roughly $f^{(k-1)}(p) \leq 0.5$.

Proof:

As $p > f(p)$ and $f(p) \in [0, 1]$,

It follows that $p > f(p) > f^{(2)}(p) > \ldots \ldots > f^{(k-1)}(p) > \ldots \ldots$

To be uniform, let $p = f^{(0)}(p)$; $f(p) = f^{(1)}(p)$; and $N_0 = 1$.

Also because

$f^{(i)}(p) + [f^{(i)}(p)]^2 + \ldots \ldots + [f^{(i)}(p)]^m = f^{(i)}(p)\{1 - [f^{(i)}(p)]^m\}/[1 - f^{(i)}(p)]$, Thus,

$f^{(0)}(p)\{1-[f^{(0)}(p)]^m\}/[1-f^{(0)}(p)]$

$\quad > f^{(1)}(p)\{1-[f^{(1)}(p)]^m\}/[1-f^{(1)}(p)]$

$\quad > \ldots \ldots$

$\quad > f^{(k-1)}(p)\{1-[f^{(k-1)}(p)]^m\}/[1-f^{(k-1)}(p)]$

$\quad > \ldots \ldots$

As $N_k = N_{k-1} f^{(k-1)}(p)\{1-[f^{(k-1)}(p)]^m\}/[1-f^{(\mathrm{k}-1)}(p)]$

$= f^{(0)}(p)\{1-[f^{(0)}(p)]^m\}/[1-f^{(0)}(p)] \times$

$f^{(1)}(p)\{1-[f^{(1)}(p)]^m\}/[1-f^{(1)}(p)] \times$

$\ldots \ldots \times$

$f^{(k-1)}(p)\{1-[f^{(k-1)}(p)]^m\}/[1-f^{(\mathrm{k}-1)}(p)]$

Given $N_k < 1$

Hence, $f^{(k-1)}(p)\{1-[f^{(k-1)}(p)]^m\}/[1-f^{(\mathrm{k}-1)}(p)] < 1$.

Otherwise, $f^{(0)}(p)\{1-[f^{(0)}(p)]^m\}/[1-f^{(0)}(p)]$,

$f^{(1)}(p)\{1-[f^{(1)}(p)]^m\}/[1-f^{(1)}(p)]$, … …,

$f^{(k-1)}(p)\{1-[f^{(k-1)}(p)]^m\}/[1-f^{(\mathrm{k}-1)}(p)]$

all are greater than 1, and N_k can not be less than 1.

Consequently,

$f^{(i-1)}(p)\{1-[f^{(i-1)}(p)]^m\}/[1-f^{(i-1)}(p)] < 1$, for any $i > k$.

Also, $1 > N_k > N_{k+1} > \ldots \ldots$, because $N_i = N_{i-1} f^{(i-1)}(p)\{1 - [f^{(i-1)}(p)]^m\}/[1 - f^{(i-1)}(p)]$

Thus,

$N = N_1 + N_2 + N_3 + \dots\dots + N_{k-1} + N_k + \dots\dots$

$= N_1 + N_2 + N_3 + \dots\dots + N_{k-1} +$

$N_{k-1} f^{(k-1)}(p)\{1 - [f^{(k-1)}(p)]^m\}/[1 - f^{(\mathrm{k}-1)}(p)] +$

$N_{k-1} f^{(k-1)}(p)\{1 - [f^{(k-1)}(p)]^m\}/[1 - f^{(\mathrm{k}-1)}(p)] \times$

$f^{(k)}(p)\{1 - [f^{(k)}(p)]^m\}/[1 - f^{(\mathrm{k})}(p)] +$

$\dots\dots$

$< N_1 + N_2 + N_3 + \dots\dots + N_{k-1} +$

$N_{k-1} f^{(k-1)}(p)\{1 - [f^{(k-1)}(p)]^m\}/[1 - f^{(\mathrm{k}-1)}(p)] +$

$N_{k-1}\{f^{(k-1)}(p)\{1 - [f^{(k-1)}(p)]^m\}/[1 - f^{(\mathrm{k}-1)}(p)]\}^2 +$

$N_{k-1}\{f^{(k-1)}(p)\{1 - [f^{(k-1)}(p)]^m\}/[1 - f^{(\mathrm{k}-1)}(p)]\}^3 +$

$\dots\dots$

$= N_1 + N_2 + N_3 + \dots\dots + N_{k-2} +$

$N_{k-1}\{f^{(k-1)}(p)\{1 - [f^{(k-1)}(p)]^m\}/\{1 - 2f^{(\mathrm{k}-1)}(p) + [f^{(k-1)}(p)]^{m+1}\}$

Therefore, the value of N is limited.

Again, as $N_{\mathrm{k}-1} + N_k + N_{k+1} + \dots\dots$ should be positive,

$N_{k-1}\{f^{(k-1)}(p)\{1 - [f^{(k-1)}(p)]^m\}/\{1 - 2f^{(\mathrm{k}-1)}(p) + [f^{(k-1)}(p)]^{m+1}\}$ should > 0.

With the effect of $[f^{(k-1)}(p)]^{m+1}$ omitted, then $1 - 2{\cdot}f^{(\mathrm{k}-1)}(p) \geq 0$.

Thus, $f^{(k-1)}(p) \leq 0.5$ □

Theorem 1 indicates that N can be limited whenever N_k is less than 1 for some k. In practice, a large number of k is still unacceptable. In the section of simulation, the k selection will be discussed. Here, one more rule should be given to complete the design.

Rule 4: *For a positive integer k, and any $p \in [0, 1]$, the shared function $f(p)$ should satisfy $p > f(p)$ and $N_k < 1$.* (4-R_4)

4.3.2 The Number of Successful Publishing

Based on conditions given by Theorem 1, the average number of publishing packets incurred by a raw incident is limited. However, it is expected that most of those publishing packets will be delivered to their termination nodes successfully during the recursive processes of log publishing. If publishing packets are simply dropped on their way before they reach their termination nodes, these packets will be of little help on log publishing, but only load extra traffics over their delivery path. In this subsection, the average number of publishing packets finally arriving at their termination nodes will be calculated. Still, the calculation is conducted round by round.

The first recursive round

On average, there are N_1 publishing packets at this round. Each of these packets at this round has the same probability $[f(p)]^m$ of reaching its termination node. Therefore, on average, $N_1[f(p)]^m$ publishing packets will arrive at their termination nodes at the first recursive round.

The second recursive round

At the second recursive round, there are N_2 publishing packets on average. Each of the packets has $[f^{(2)}(p)]^m$ chance to arrive at its termination nodes. Thus, on the average sense, $N_2[f^{(2)}(p)]^m$ publishing packets will arrive at their termination nodes at the second recursive round.

The nth recursive round

At the *n*th recursive round, we have N_n publishing packets on average. Each of the packets has $[f^{(n)}(p)]^m$ chance to arrive at their termination nodes. Therefore, on average, $N_n[f^{(n)}(p)]^m$ publishing packets will arrive at their termination nodes at the *n*th recursive round.

4.3.3 Optimization

Above all, the total number of successful publishing packets on average should be $N_1[f(p)]^m + N_2[f^{(2)}(p)]^m + \dots \dots + N_n[f^{(n)}(p)]^m + \dots ....$ Thus, to make most publishing packets useful or to deliver them successfully, a function $f(p)$ should be selected in order to satisfy the following criterion.

Criterion 1: *To optimize PD2, $f(p)$ should be selected to maximize the ratio of*

$$\frac{N_1 \cdot [f(p)]^m + N_2 \cdot [f^{(2)}(p)]^m + \dots\dots + N_n \cdot [f^{(n)}(p)]^m + \dots\dots}{N_1 + N_2 + N_3 + \dots\dots + N_n + \dots\dots} \tag{4-C$_1$}$$

Because all $f(p), f^{(2)}(p), \dots \dots, f^{(n)}(p), \dots \dots$, are positive and less than 1, it is clear that the above ratio will reach its maximum if all $f(p), f^{(2)}(p), \dots \dots, f^{(n)}(p), \dots \dots$, are close to 1. However, if all $f(p), f^{(2)}(p), \dots \dots, f^{(n)}(p), \dots \dots$, are close to 1, the average number of total publishing packets N would be too large to accept. Thus, there is a tradeoff with regard to optimizing the performance of PD2.

Based on the (4-R_1) through (4-R_4) and (4-C_1), the scheme of PD2 can be designed. The function $f(p)$ can be constructed by following the Rule 4 to limit the value of N, and by following (4-C_1) to gain better performance.

4.3.4 An Example of $f(p)$ Construction

In this section, linear function $f(p) = a \cdot p$ is investigated. The task is to determine the optimal value of a for any $p \in [0, 1]$, by following (4-R_4) and (4-C_1). Here, the hop count m is set at 4. As $f(p) = ap$; $f^{(2)}(p) = a \cdot (a \cdot p) = a^2 p$; ……; $f^{(k-1)}(p) = a^{(k-1)} p$; …… By following (4-$R_4$) and set $p = 1$, the results of the calculation are shown in Table 4-1.

Table 4-1. An example of publishing rate

	$a = 0.4$	$a = 0.6$	$a = 0.8$
N_1	3	3	3
N_2	1.9488	3.9168	7.0848
N_3	0.3710	2.1662	10.4821
N_4		0.5955	10.2419
N_5			6.9055
N_6			3.3268
N_7			1.1764
N_8			0.3116

As shown in Table 4-1, to satisfy, when $a = 0.4$, k should be 3; when $a = 0.6$, k should be 4; when $a = 0.8$, k should be 8. Similarly, $f(p)$ can also be constructed in more complex ways, such as $ap^2 + bp + c$, αp^β, etc. All efforts regarding the $f(p)$ construction are to improve the performance of the PD2 design.

4.4 Simulation

Using an NS-2 simulator, simulations run within a typical MANET environment. The specifications of the MANET mainly follow the model given in section 4.2. The area of the MANET is set to be $1000 \times 1000\text{m}^2$. Within the area, nodes are settled at cross-points of a uniform square grid, with the grid spacing d = 200m. The transmission radius of each node is set to be r = 250m, and AODV (Ad hoc On-Demand Distance Vector) is selected as the routing protocol.

The goal of simulations is to investigate the performance of PD2, when $f(p)$ is defined as $f(p) = ap$ and m is set at 4 for all recursive rounds. During the theoretic analysis, the average number of publishing packets is calculated. In this section, distributions of the total number of publishing packets are presented, as are distributions of the number of successfully published packets. A UDP packet delivery initiates the whole process, and its delivery rate p is set to be 1 and 0.9, respectively.

Figures 4-2 and 4-3 show distributions when the delivery rate is $p = 1$. As shown in the two figures, the distribution functions shift towards the right side, when the factor a increases from 0.4 to 0.8. As indicated in the Figure 4-2, the total number of publishing packets are approximately 10 or 11 when $a = 0.4$; roughly 14 or 15 when $a = 0.6$; and about 30 or 31 when $a = 0.8$.

Compared with the other two cases, many publishing packets are incurred when $a = 0.8$. From Figure 4-3, when $a = 0.4$, the probability of successful publishing is less than

0.1; when *a* = 0.6, the probability of successful publishing is about 1/4; and when *a* = 0.8, the probability of successful publishing is about 0.5. Compared with the other two cases, the probability of successful publishing is very small when *a* = 0.4. Based on the observation of these two figures, *a* = 0.6 is optimal for $f(p)$.

The distributions when the delivery rate *p* = 0.9 are presented in Figures 4-4, 4-5. Through observation, a similar conclusion can also be drawn; specifically, *a* = 0.6 is a better choice than are the other two cases (when *a* = 0.4 and *a* = 0.8).

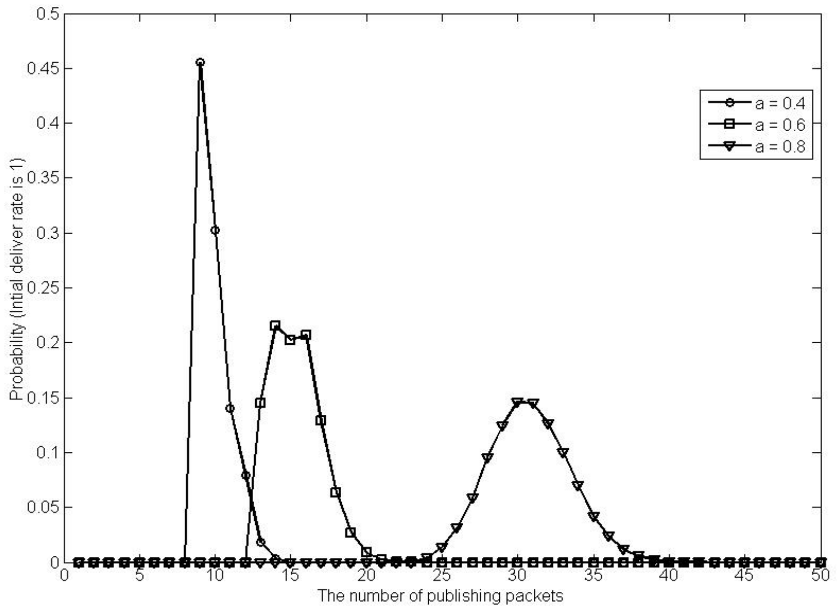

Figure 4-2. Probability distribution of publishing packets when *p* = 1

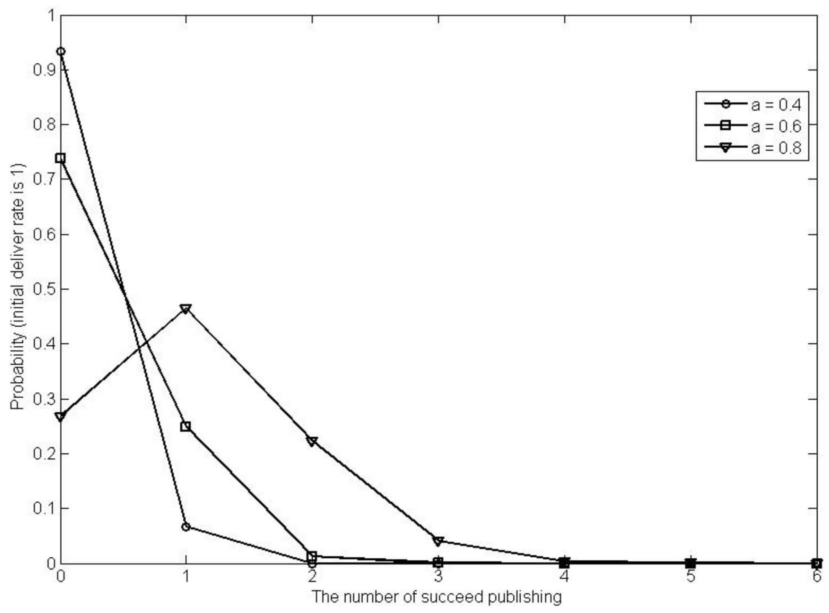

Figure 4-3. Probability distribution of successful publishing when $p = 1$

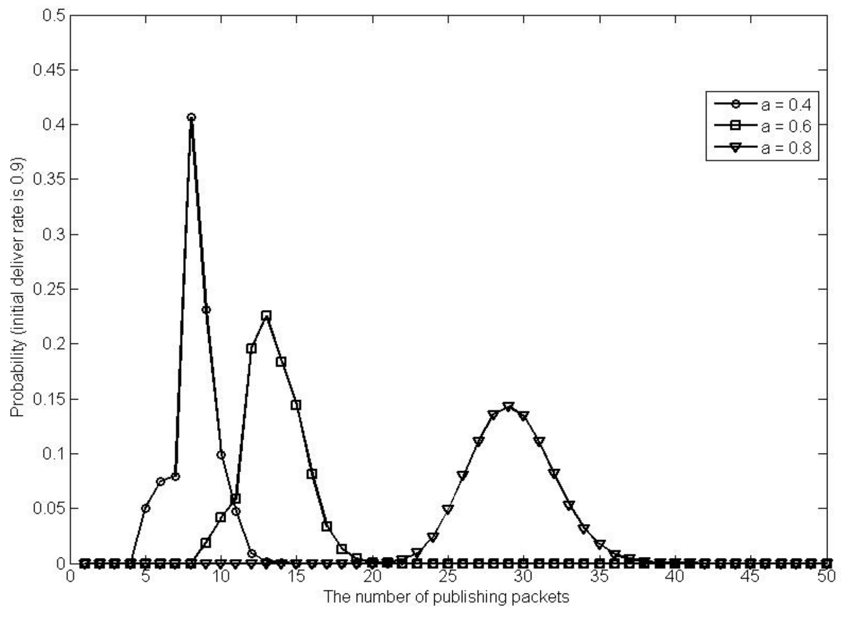

Figure 4-4. Probability distribution of publishing packets when $p = 0.9$

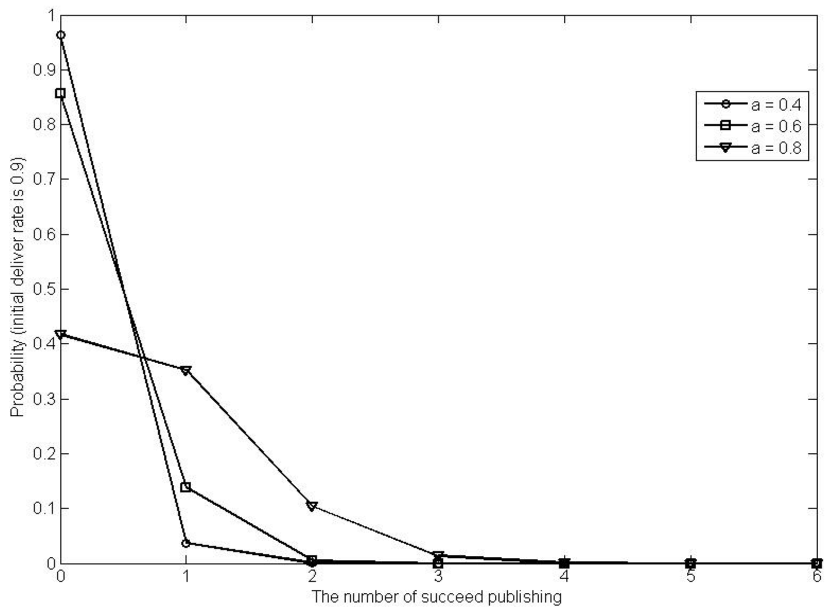

Figure 4-5. Probability distribution of successful publishing when $p = 0.9$

Summary

The technology of DHT has been introduced, in terms of incident publishing. This technology aims to organize log files into a distributed database. In addition, the problem of recursive publishing is put forward, and the solution regarding to this recursion problem has been given. This solution limits the total communication cost of log publishing.

In this chapter, a general mobile network is taken as the model of the research. In the next two chapters, the incident publishing would be studied in some specific networks, and the objective of the research is still to reduce the communication cost of incident publishing.

Chapter 5 Optimizing Location Publishing over Cellular Networks

With a general model of mobile networks, incident publishing has been studied in the last chapter. In this chapter, incident publishing is still the theme, but the study is targeted on cellular networks, which are the most popular mobile networks. In addition, the research issue of incident publishing narrows down to the *location* publishing. The objective of the study is to optimize the accumulative cost for the process of *location* publishing, at the mean time, to facilitate tracking the movement of mobile subscribers.

In this chapter, the structure of “anchor chain” [Bej03] is followed. According to the “anchor chain” design, a mobile subscriber only needs to publish its location on the anchor. In this way, the communication cost of *location* publishing would be decreased. In addition, the built-in memory of a subscriber is utilized to record the chain structure of this subscriber. As a result, the subscriber would carry the record of its chain structure along its moving path. When the subscriber moves in a region served by a new VLR, it would submit the record of its chain structure to this new VLR. With this submitted record, the new VLR would construct a new anchor chain for the subscriber. After that, this new chain is downloaded to the subscriber to update its chain record.

With the “anchor chain” design, an investigator no longer needs to reference the HLR every time, in order to track the subscriber’s movement. In fact, the investigator only needs to contact the nearest anchor, and this anchor works as an agent to obtain the subscriber’s current *location*. When the nearest anchor is discarded by the chain update, the investigator may search backward along the original chain, or simply contact the HLR, in regards to finding active anchors. By referencing the nearest anchor rather than the HLR, the investigator may spot the mobile subscriber fast.

This chapter is organized as follows: in Section 5.1, the problem of *location* publishing is described; in Section 5.2, the research model is formulated, and the design is presented; Performance complexity of the design is analyzed in Section 5.3; Section 5.4 examines simulations to verify the efficiency of the design.

5.1 Problem Description

In cellular networks, mobile subscribers should always publish their *location*s when roaming. The process of this publishing is generally called “*location* registration” or “*location* binding.” The moving path of a mobile subscriber can always be traced by referencing its published *location*s.

Research efforts on optimizing the process of *location* registration have never been halted since the inception of mobile communication. According to the standard IS-41, mobile subscribers should always register their current *location*s on their HLRs directly. However, when a mobile subscriber is far away from its HLR, or when the mobile subscriber changes its *location* frequently, the cost of its *location* publishing is huge. According to the standard IS-41, the accumulative publishing cost of a mobile subscriber can be $O(M^2)$ in the worst scenarios. Here, M denotes the distance that a mobile subscriber has moved.

In addition, to track the movement of a mobile subscriber, the investigator must reference the HLR of this subscriber. Mostly, the subscriber and the investigator are not far away from each other. Therefore, it would seem a waste to the investigator to reference the subscriber's HLR, whenever the subscriber moves.

5.2 Model and Algorithm

This section is divided into five subsections. The first sub-section states assumptions and concepts. The second sub-section full describes the model on which the study would be based. In the third and fourth sub-section, two designs, i.e. static scheme and dynamic scheme, are specified respectively.

5.2.1 Assumptions and Concepts

5.2.1.1 Assumptions

a) Communication cost between any two nodes refers to the communication cost along the shortest routing path of these two nodes.

b) All paths are bi-directional, with the same communication cost for each direction.

c) Each mobile subscriber has one HLR.

5.2.1.2 Concepts

a) Location Information Value ($V_{\text{L.I.}}$):

The metric, $V_{\text{L.I.}}$ (Location Information Value), derives from the communication cost. As shown in Figure 5-1, there is an association (i.e. chain links) from HLR to VLR_1, and from VLR_1 to VLR_2. Then, the $V_{\text{L.I.}}$ of VLR_2 is the communication cost from

VLR_1 to VLR_2; the $V_{\mathrm{L.I.}}$ of VLR_1 is the communication cost from HLR to VLR_1; and the $V_{\mathrm{L.I.}}$ of HLR is 0.

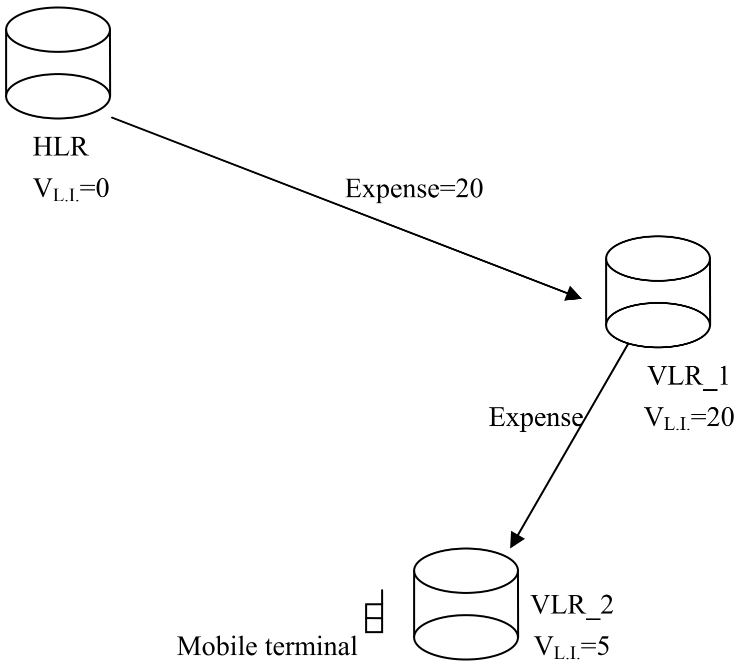

Figure 5-1. Example I of location information value

It is important to note that the values of $V_{\mathrm{L.I.}}$ are related to the direction of the association. As shown in Figure 5-1, the direction of the association is HLR→VLR_1→VLR_2. In such case, the values of $V_{\mathrm{L.I.}}$ are 0, 20, and 5, which are corresponding to HLR, VLR_1, and VLR_2 respectively. If the direction of the association is reversed, as shown in Figure 5-2, the association of these three nodes becomes VLR_2→VLR_1→HLR. In this case, $V_{\mathrm{L.I.}}$ of VLR_2 should be 0, $V_{\mathrm{L.I.}}$ of VLR_1 should be 5, and $V_{\mathrm{L.I.}}$ of HLR should be 20.

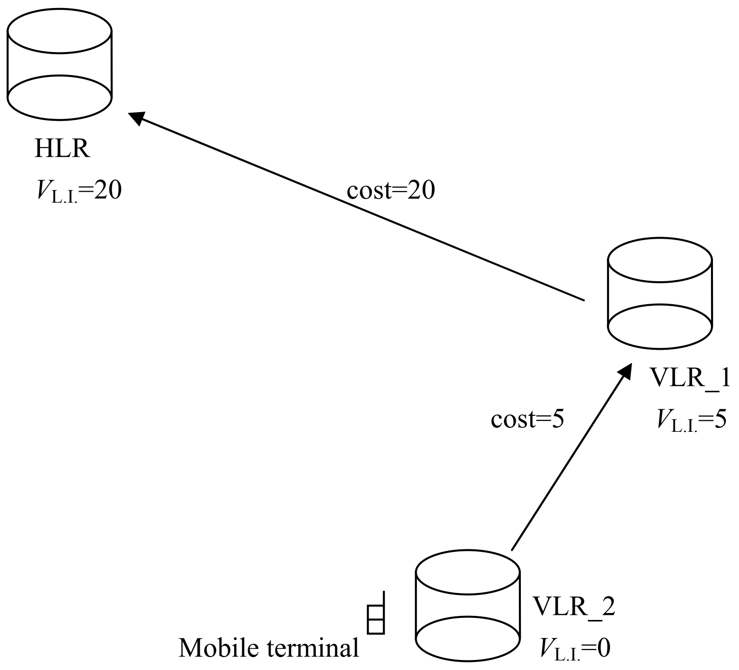

Figure 5-2. Example II of location information value

b) Anchor chain structure:

Table 5-1. An example of anchor chain structure

Seq_No.=0,	ID=HLR,	$V_{L.I.}$=0;
Seq_No.=1,	ID=VLR_1,	$V_{L.I.}$=20;
Seq_No.=2,	ID=VLR_2,	$V_{L.I.}$=5.

The anchor chain structure includes ID numbers and values of $V_{L.I.}$ of the anchor nodes along the chain. In addition, the anchor chain structure should include the sequence numbers of the anchor nodes. The direction of an anchor chain goes from the HLR to the serving VLR. The $V_{L.I.}$ of each anchor node is the communication cost from the previous neighbor anchor to itself. For the anchor chain shown in Figure 5-1, the anchor chain structure is listed in Table 5-1.

One thing that needs to be concerned is the generation of $V_{\mathrm{L.I.}}$s. All anchors of a chain are nodes that the mobile subscriber has visited. Therefore, the $V_{\mathrm{L.I.}}$ of an anchor node can be obtained by referencing the node's routing table, during the time that the mobile subscriber is visiting.

5.2.2 The Model

In a GSM network, every node functions as an LR (*location* register). For a mobile subscriber, one of these LRs is the subscriber's HLR, and all the other LRs are the subscriber's VLRs. Each mobile subscriber possesses a built-in memory and a timer. The built-in memory would be used to record the subscriber's chain structure.

When a mobile subscriber enters a new *location*, its anchor chain structure will be submitted to the LR that serves this new *location*. After that, a process of chain update is invoked at the serving LR. The output of the chain update is a new chain structure. This new anchor chain would also be downloaded to the mobile subscriber, in order to update its original chain structure. When a call request arrives at the subscriber, the current serving LR will publish its *location* on the mobile subscriber's HLR directly. As a result of the call request, the anchor chain of the mobile subscriber only includes two nodes, the subscriber's HLR and the subscriber's serving LR.

When a mobile subscriber moves out of its home region, it will register on the local LR, and this local LR will bind to the subscriber's HLR directly. At this time, the anchor chain of the subscriber (i.e. from HLR to the local LR) consists of two nodes. After that, the mobile subscriber continues its movement and enters into a new area served by a new LR. The mobile subscriber registers on this new LR. Now, there are two options to update the anchor chain, as shown in Figure 5-3. One option is to construct the anchor chain from HLR to the new LR directly; the other option is to construct the anchor chain from HLR to the original local LR (original anchor), then to the new LR. The new anchor chain of the mobile subscriber must be selected from these two options.

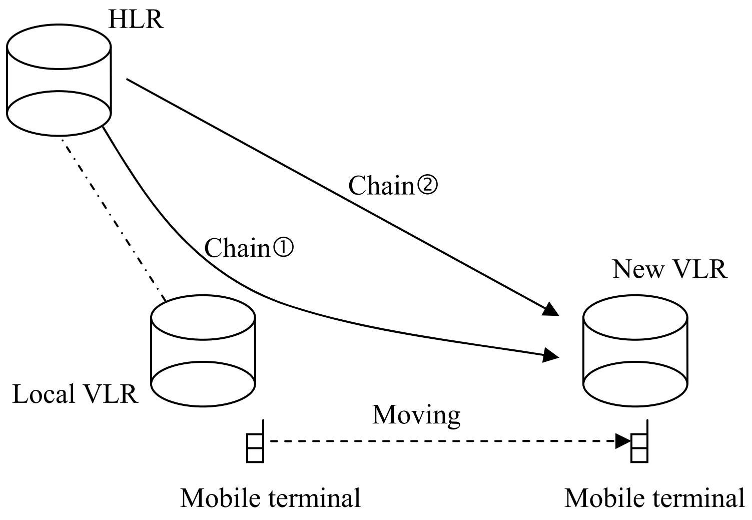

Figure 5-3. An example of chain update

Based on the discussion above, the problem of the chain update can be transferred to the problem of selecting a new anchor chain from the possible options. In the next two sub-sections, two algorithms of the anchor chain update are designed. They are static algorithm and dynamic algorithm under different assumptions.

5.2.3 Static Algorithm

Assumption: *A phone call request must arrive during the time that the subscriber is at the new LR* (5-A_S)

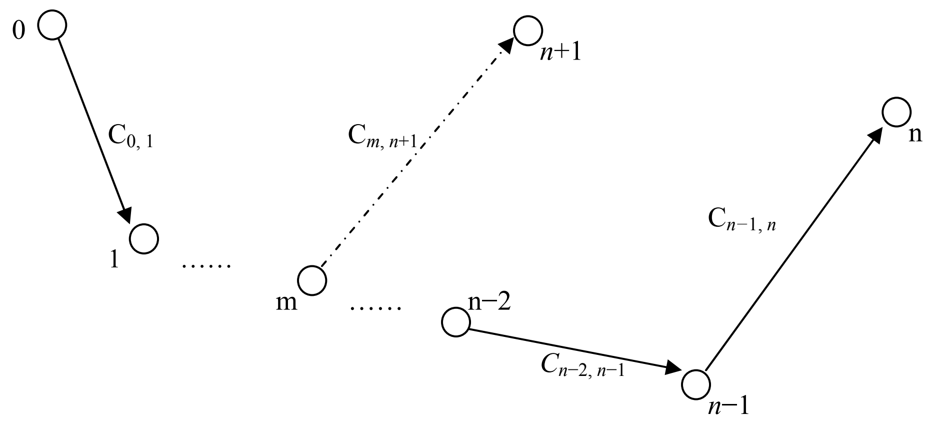

Figure 5-4. Anchor chain and its update

Anchor nodes along the chain are indexed. The HLR, which is always the origin point of an anchor chain, is indexed by ‘0’, and the index numbers of anchors increase along the chain. Let the index of the serving LR be ‘*n*’, and the new LR that the subscriber just moved in be ‘*n*+1’. An anchor with index *m* ($0 \le m \le n$) would be selected to form the new chain that is 0, 1 … *m*, *n*+1. In this way, the communication cost of *location* publishing equals the communication cost from anchor *m* to anchor *n*+1, denoted by $C_{m,n+1}$. The anchor chain and indexes of anchors are shown in Figure 5-4. The call tracking cost is $\sum_{k=1\ldots m} C_{k-1,k} + m \cdot \mathrm{E}[D] + C_{m,n+1}$. Here, E[*D*] refers to the cost of average delay at each anchor. Therefore, the optimal criterion of the anchor chain update should be finding the anchor *m* from the entire chain, in order to minimize the communication cost of *location* publishing and call tracking.

$$\min \{\sum_{k=1}^{m} C_{k-1,k} + m\cdot \mathrm{E}[D] + 2C_{m,n+1}\}|_m,\ m = 0, 1, \ldots, n \qquad \text{(5-C}_\mathrm{S}\text{)}$$

Let $\sum_{k=1\ldots m} C_{k-1,k} + m\cdot \mathrm{E}[D] + 2C_{m,n+1} = \text{Opt-S}(m)$, The static algorithm for the anchor chain update is listed in Table 5-2.

Table 5-2. Static algorithm of chain update

1.	Let $i = 0, j = 0$;
2.	If Opt-S(i) > Opt-S(j) then $i = j$; Otherwise, do nothing;
3.	$j = j + 1$;
4.	If $j = n + 1$, then go to step 6;
5.	Go to step 2;
6.	Node i is optimal access node;
7.	Assign new $\mathrm{V}_{\mathrm{L.I.}}$ to node i;
8.	Download new chain structure to subscriber

Following the static algorithm, a property of the anchor chains is presented. In brief, this property states that the "length" of an anchor chain is up-bounded. This property will be useful in the Section 5.3 where the performance complexity of the design is analyzed.

Property I: Given an anchor chain of a mobile subscriber, which is updated by following the static algorithm, any two anchors of the anchor chain with their index (i, j) and $j < i$, will always have the relationship 5- E_S hold. Here, E[D] equals to zero.

$$C_{j,i} \le \sum_{k=j+1}^{i} C_{k-1,k} \le 2C_{j,i}\,. \qquad \text{(5-E}_\mathrm{S}\text{)}$$

Proof:

There are two anchors (i, j) on the chain, and $j < i$, as shown in the Figure 5-5. Assuming that the subscriber moves back to node i at the next step, by following the static algorithm, there is a relationship:

$$2C_{i,i} + \sum_{k=1}^{i} C_{k-1,k} \leq 2 \cdot C_{i-1,i} + \sum_{k=1}^{i-1} C_{k-1,k} \leq 2C_{j,i} + \sum_{k=1}^{j} C_{k-1,k}$$

For $C_{i,i} = 0$, then

$$\sum_{k=j+1}^{i} C_{k-1,k} \leq 2C_{j,i} \ .$$

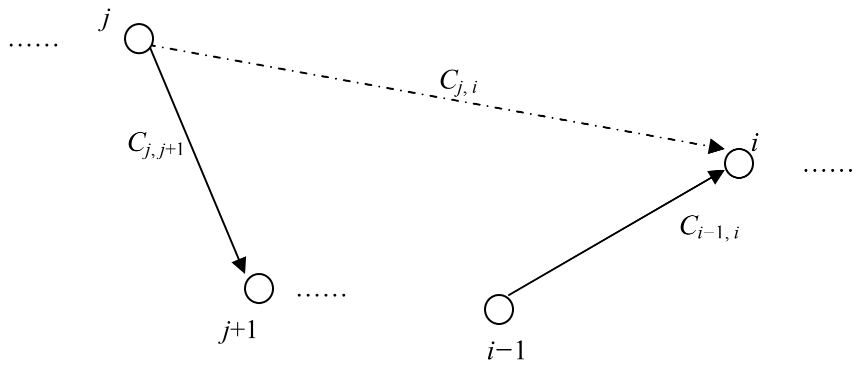

Figure 5-5. Chain structure between node i and j

Because the shortest routing path is defined over a network topology space, the network can then be treated as a normed space [Kol99]. Therefore, the triangle inequality of a normed space should be hold, and the result is

$$C_{j,i} \leq \sum_{k=j+1}^{i} C_{k-1,k} \ . \qquad \square$$

5.2.4 Dynamic Algorithm

Assumption: *The arrival of the phone call request is a Poisson process.* (5-A_D)

The anchor chain is shown in Figure 5-4. The mobile subscriber just moves from anchor n to anchor n+1. Anchor m is selected to form the new chain, which is 0, 1 … m, n+1. Therefore, the cost of *location* publishing is $C_{m,\, n+1}$, and the cost of possible call tracking is $\sum_{k=1\ldots m} C_{k-1,\, k} + m\mathrm{E}[D]+C_{m,\, n+1}$.

Let X_l be the time interval between $(l-1)^{\text{th}}$ phone call arrival and l^{th} phone call arrival, then $P\{X_l \le t\} = 1-e^{-\lambda t}$. Let μ be CMR (Call to Mobility Ratio); c denotes the average number of phone call arrivals during the time period T; and m denotes the average number of movements during the same time period T. Then, $(T/m) = (T\mu/c) = (\mu/\lambda)$ is the average time period that a mobile subscriber stays at a *location*. Let t_0 denote the time point of the beginning of the last phone call; t_1 denotes the time point of the end of the last phone call; t_2 denotes the time point of the last movement. Here, $t_2 > t_1$. Otherwise, it would be the cases of call handover and would not be included in this research.

During the time period $(t_2,\, t_2+\mu/\lambda]$, the probability of the next phone call arrival is calculated as follow. Because $P\{X_l \le t_2+\mu/\lambda-t_0\} = 1-\exp[-\lambda(t_2\text{-}t_0)-\mu]$, and no other call arrives during the time period $(t_0,\, t_2]$, the probability that a call arrives during the time period $(t_2,\, t_2+\mu/\lambda]$ is still $1-\exp[-\lambda(t_2\text{-}t_0)-\mu]$. Then, the average cost of call tracking during the time period $(t_2,\, t_2+\mu/\lambda]$ is:

$$\{\sum_{k=1}^{m} C_{k-1,\, k} + m\cdot\mathrm{E}[D]+C_{m,\, n+1}\}\{1-\exp[-\lambda(t_2\text{-}t_0)-\mu]\}. \qquad (5\text{-}E_D)$$

Let Opt-D(m) = $\{\sum_{k=1\ldots m} C_{k-1,\,k} + m\cdot\mathrm{E}[D] + C_{m,\,n+1}\}\{1-\exp[-\lambda(t_2\text{-}t_0)-\mu]\} + C_{m,\,n+1}$, the optimization criterion is to search along the anchor chain to find the minimum Opt-D(m).

$$\min\ \{\text{Opt-D}(m)\},\ m = 0, 1, \ldots, n. \qquad (5\text{-}C_D)$$

As shown in Table 5-3, the dynamic algorithm is similar to the static algorithm, except that Opt-D(m) substitutes for Opt-S(m). Also, there is a similar property to up-bound the length of anchor chains, which is updated by following the dynamic algorithm.

Table 5-3. Dynamic algorithm of chain update

```
1.  Let $i = 0, j = 0$;
2.  If Opt-D($i$) > Opt-D($j$) then $i = j$;
    Otherwise, do nothing;
3.  $j = j + 1$;
4.  If $j = n + 1$, then go to step 6;
5.  Go to step 2;
6.  Node $i$ is optimal access node;
7.  Assign new $V_{L.I.}$ to node $i$;
8.  Download new chain structure to
    subscriber
```

For both static and dynamic algorithms, some anchor nodes may be discarded from the chain after a chain update. Especially for the cases that a discarded anchor is the agent of an investigation, the movement tracking would be disturbed. However, the investigator can still find an active anchor by searching along the original chain. In this design, anchors are required to log the anchor chain periodically. Whenever movement tracking is stopped, the investigator should request these logged chains from the agent.

5.3 Performance Analysis

Let $D(u, v)$ be the distance between nodes (LRs) of u and v, and $C(u, v)$ be the communication cost between them. There is a relationship $C(u, v) \le kD(u, v)$, which is widely accepted [Bej03]. Here, k is a constant. This inequality implies an equivalent relationship between $C(u, v)$ and $D(u, v)$, Since $0D(u, v) \le C(u, v) \le c{\cdot}D(u, v)$. Therefore, $C(u, v)$ and $D(u, v)$ are exchangeable, in terms of performance analysis. In this section, only $D(u, v)$ is used.

In this section, the performance complexity is analyzed in respect to the static algorithm. The extension of the analysis to the dynamic algorithm is trivial. In addition, only the communication cost of *location* publishing is considered. For the accumulative cost of "*location* publishing," three worst cases exist. They are discussed in the following five sub-sections.

5.3.1 Case I

As shown in Figure 5-6, a mobile subscriber starts its travel from its HLR. At the first step, the subscriber moves from its HLR to node 1 (maybe by aircraft). Then, M, which is the distance the subscriber has moved, equals L. Also, *Bind*, which is the accumulative *location* publishing cost, equals L.

After that, the mobile subscriber moves to node 2. When the mobile subscriber is at node 2, it has two options to publish its current *location* on node 1, or on its HLR. By following the static algorithm, the mobile subscriber would publish its *location* on the node 1, if $2D(1,2) + L < 2[L - D(1,2)]$. Otherwise, the mobile subscriber would publish its *location* on the HLR. Therefore, $0.25L$ is a critical value for the second step $D(1,2)$. Because $2D(1,2) + L = 2[L - D(1,2)]$ will be hold when $D(1,2) = 0.25L$. If $D(1,2) < 0.25L$, the mobile subscriber will bind to node 1. In this case, $Bind = D(1,2) + L$ and $M = D(1,2) + L$, which is not the worst case. If $D(1,2) > 0.25L$, the mobile subscriber would publish its *location* on HLR. In this case, $M = L + D(1,2)$ and $Bind = 2L - D(1,2)$. The ratio of $M/Bind$ reaches its minimum 5/7, when $D(1,2) = 0.25L$.

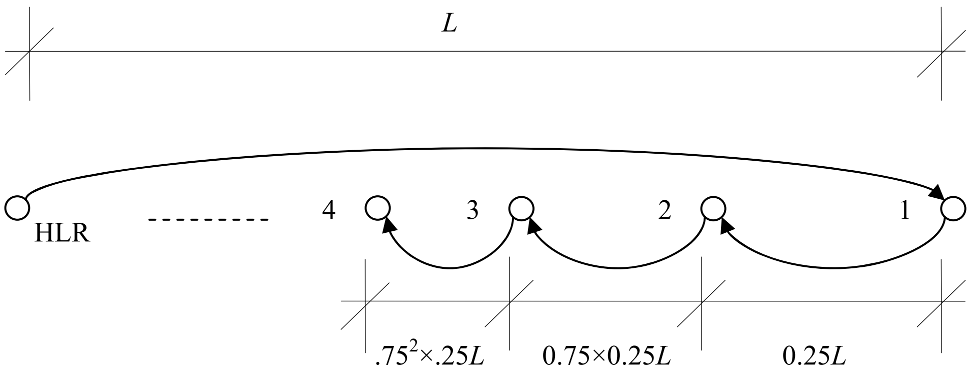

Figure 5-6. Chain updates of the worst case I

Similar analyses can be done in the following steps, such as the step from node 2 to node 3, from node 3 to node 4, …, until the mobile subscriber goes back to its HLR (after infinite steps). Therefore, in the worst case I,

$$M = L + 0.25L + 0.25(1-0.25)L + 0.25(1-0.25)^2L + \ldots = 2L$$

$$Bind = L + (1-0.25)L + (1-0.25)^2L + (1-0.25)^3L + \ldots = 4L$$

$$Bind = 2M \qquad (5\text{-}E_1)$$

5.3.2 Case II

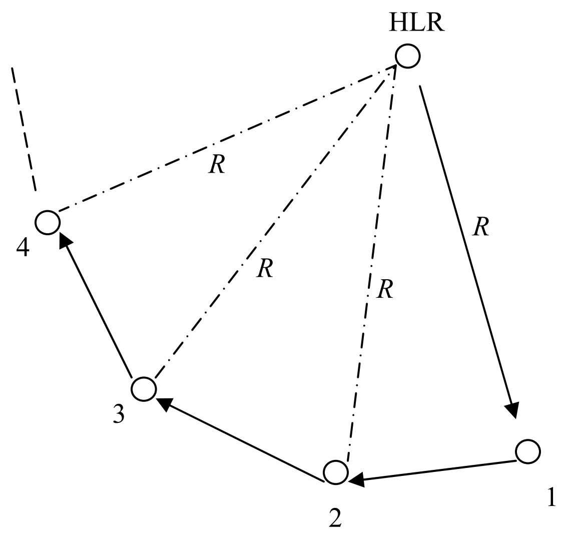

Figure 5-7. Chain updates of the worst case II

As shown in Figure 5-7, a mobile subscriber starts from HLR. After the first step, the mobile subscriber moves along a circle. In addition, the HLR is at the centre of the circle, and the radius is R. When the mobile subscriber moves to node 2, the critical condition is $2R = R + 2D(1,2)$. By following the similar logic presented in case I, the critical distance for $D(1,2)$ is $0.5R$. Now, the mobile subscriber is at node 2. If $D(1,2) > 0.5R$, the mobile subscriber would publish its *location* on the HLR. In this case, $M = R + D(1,2) = 1.5R$; $Bind = 2R$. The ratio of ($M/Bind$) reaches its minimum 3/4 when $D(1,2) = 0.5R$. Similar analyses can be processed in the following steps, such as the step from node 2 to node 3, from node 3 to node 4, and so on. After N steps,

$M = R + 0.5RN$; Bind $= R + RN$.

Therefore, in the worst case II, $Bind = 2M$ when $N \to \infty$, (5-E_2)

5.3.3 Preparation I for Case III

As shown in Figure 5-8, only the movement and the publishing cost after node *C* are taken into account. The mobile subscriber moves from node *B* to node *A* at the last step, and publish its *location* on node *C*. Let the distance from *C* to *B* along the anchor chain (which is a curvy path) be $D(C\sim B) = \beta D(C, B)$. According to the property I (up-bound property) in Section 5.2, it is known that $1 \leq \beta \leq 2$.

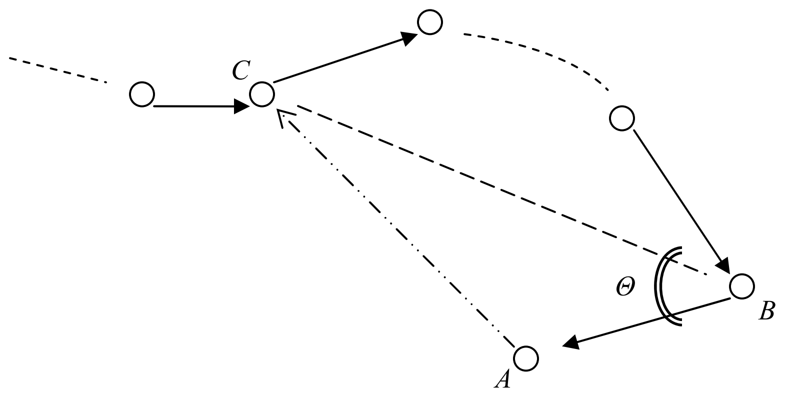

Figure 5-8. Chain updates for preparation I of the worst case III

To be one of the worst cases, node *A* should be located at the critical point, which means that

$$2D(A, C) = \beta D(C, B) + 2D(A, B) \qquad \ldots\text{(5-Ea)}$$

The network can be mapped into Euclidean spaces, because it is assigned with a *distance metric* (i.e. the shortest routing path). Therefore, $D(A, C)$, $D(C, B)$, and $D(A, B)$ should abide by the Law of Cosines. Therefore,

$$D^2(A, C) = D^2(C, B) + D^2(A, B) - 2D(C, B)D(A, B)\cos\Theta \qquad \ldots\text{(5-Eb)}$$

From equations (5-Ea) and (5-Eb),

$D(A, B) = [(1 - 0.25\beta^2)/(\beta + 2\cos\Theta)]D(C, B)$, and, (5-Ec)

$D(A, C) = [(0.25\,\beta^2 + \beta\cos\Theta + 1)/(\beta + 2\cos\Theta)]D(C, B)$ (5-Ed)

For the worst case, the subscriber publishes its *location* on node *C*. Then,

$M = D(C{\sim}B) + D(A, B)$; $Bind = D(C{\sim}B) + D(A, C)$, and,

$M/Bind = (3\beta^2+8\beta\cos\Theta+4)/(5\beta^2+12\beta\cos\Theta+4)$ (5-Ee)

Based on the results above, no matter how much Θ is, the ratio would get its minimum 2/3 when $\beta = 2$. In this scenario, $D(A, B) = 0$ and $D(A, C) = D(C, B) = 0.5D(C{\sim}B)$.

5.3.4 Preparation II for Case III

As shown in Figure 5-9, this situation is almost the same as that described in subsection 5.3.3, except the link from anchor *D* to anchor *E*. From anchor *D* to anchor *E*, the mobile subscriber does not move one step, but goes multiple steps along a curvy path. When the mobile subscriber travels along this curvy path, its anchor chain also goes along this curvy path. Until the mobile subscriber reaches node *E*, the chain is updated and the mobile subscriber publishes its *location* on node *D*. As a result, the anchor chain of the mobile subscriber goes from node *D* to node *E* directly (i.e. along the shortest routing path between these two nodes). In fact, what happened from node *D* to node *E* is the same as that from node *A* to node *C* presented in Subsection 5.3.3.

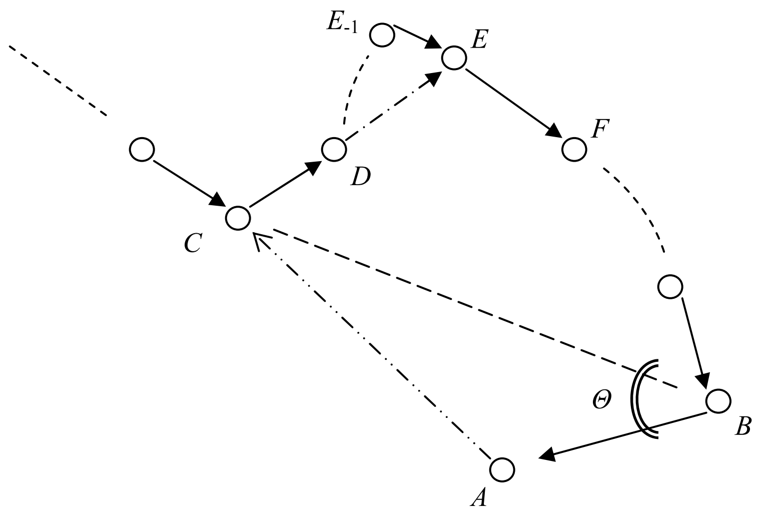

Figure 5-9. Chain update for preparation II of the worst case III

Before the chain link from node *D* to node *E* is set up, the subscriber moves from *D* to *E* along a curvy path with multiple steps. E_{-1} is the node from which the subscriber moves to node *E*. Let $D(D\sim E_{-1})$ be the distance from *D* to E_{-1} along the curvy path, and $D(D\sim E_{-1}) = \beta_{DE}D(D, E_{-1})$. According to property I in Section 5.2, $1 \leq \beta_{DE} \leq 2$.

Now, the mobile subscriber moves from node *A* to node *B*. The accumulative distance the mobile subscriber moves after node *C* is:

$M = D(C\sim B) + D(A, B) - D(D, E) + D(D\sim E_{-1}) + D(E_{-1}, E)$

The accumulative communication cost of *location* publishing after node *C* is:

$Bind = D(C\sim B) + D(A, C) + D(D\sim E_{-1})$

By following the similar analysis logic performed in preparation I, the minimum ratio $M/Bind$ can only be approached when $\beta = 2$ and $\beta_{DE} = 2$.

5.3.5 Case III

As shown in Figure 5-10, the solid line represents the moving path of the mobile subscriber. All broken lines denote links that are solely caused by the chain update. In fact, case III is a generalized situation of the subsections 5.3.3 and 5.3.4.

Let the length of the solid line be L. Based on results of preparation I and II, in the worst case, the length of the broken line ② should be $0.5L$, the length of the broken line ③ is 0.5^2L, and the length of the broken line ④ is 0.5^3L, and so on. Hence:

$M = L$, and, $Bind = L + 0.5L + 0.5^2L + 0.5^3L + \ldots = 2L$.

Again, for the worst case III, $Bind = 2M$. (5-E3)

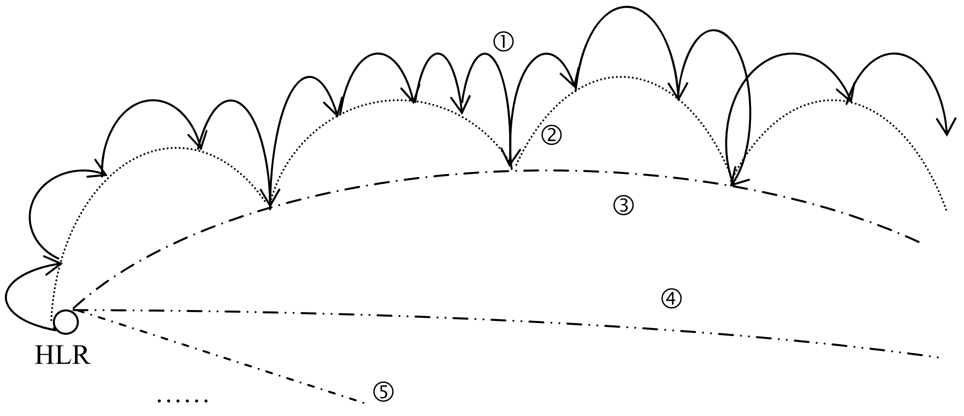

Figure 5-10. Chain update of the worst case III

Any movement and *location* publishing can be bound by these 3 worst cases. Thus, the ratio $M/Bind \leq 0.5$, and the cost of *location* publishing is within the order of O(M).

5.4 Simulation

Simulations are conducted within an area of 1000×1000 square miles. In the area, two thousand LRs are scattered with a uniform distribution. The geographic distance is used as the communication cost between nodes. In addition, twenty mobile subscribers move randomly. Their HLRs are scattered in the area with a uniform distribution. For the first scenario, CMR varies from 2 to 20. The accumulative cost of *location* publishing for each strategy is divided by the accumulative cost for the standard IS-41. "Anchor Chain" represents the strategy proposed in [Bej03]. As shown in Figure 5-11, along with the increment of CMR, the performances of static and dynamic strategies are getting better. Moreover, the performances of these two (i.e. static and dynamic strategies) are better than those of the "anchor chain" strategy proposed in [Bej03].

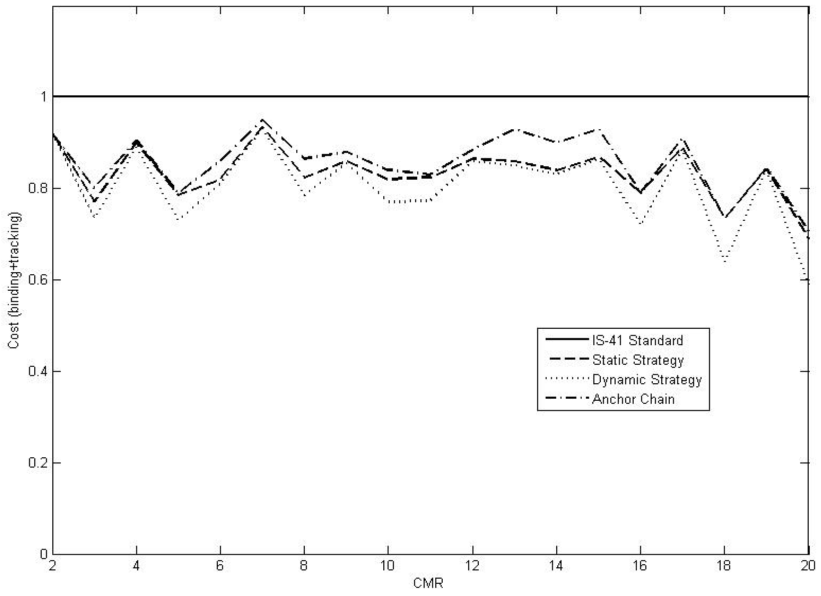

Figure 5-11. Performance comparison of different strategies

In the next simulation, the ratios (*M*/*Bind*) of these four strategies are considered. The moving step for each subscriber is limited to three hundred miles. As shown in Figure 5-12, it is evident that the slopes of both static and dynamic strategies are lower than that of the "anchor chain" strategy proposed in [Bej03].

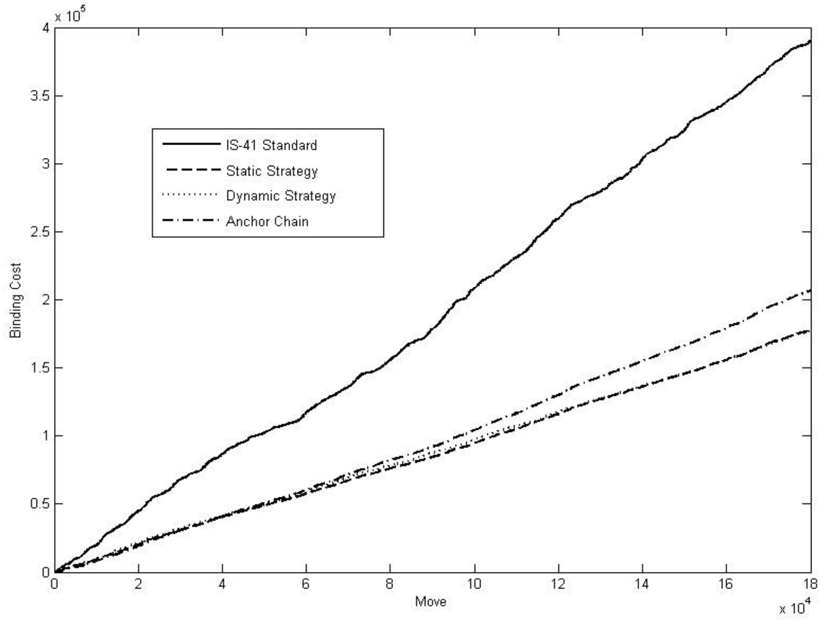

Figure 5-12. Relationship of binding cost and movement

Summary

In terms of *location* publishing, the design of "anchor chain" [Bej03] has been improved. Additionally, the accumulative cost of *location* publishing is minimized. More importantly, the design presented in the chapter may greatly reduce the time of movement tracking. As a result, policeman can quickly spot the criminal subscribers. Incident publishing (or *location* publishing) has been researched in cellular networks. According to the chapter, the cost of incident publishing can be reduced greatly, when some structure (i.e. anchor chain) is introduced in the process of incident publishing. In the next chapter, the technology of “in-network processing” would be introduced for the process of incident publishing, with regards of reducing the cost of the physical incident publishing.

Chapter 6 Physical Incident Publishing over Wireless Sensor Networks

In this chapter, the incident publishing is studied in the context of wireless sensor networks, which is another popular network model of the current research. Specifically, the incident refers to the physical incident that a sensor node has sensed. In addition, the sink node is the termination of each process of incident publishing. The objective of this research is still to reduce the communication cost of incident publishing.

6.1 Problem Description

For wireless sensor networks (WSNs), source nodes (i.e. the sensor nodes with sensory date) should have their sensory data (indicating physical incidents) published at the sink node. In most cases, the sink node is far away from the source nodes. This implies that it is costly to publish sensory data from every source node to the sink node. Thereafter, the technology of "in-network processing" is developed [Hei01]. According to this technology, the process of publishing sensory data at the sink node is divided into two phases. In the first phase, sensory data should be converged to some local nodes, and aggregated there. In the second phase, these aggregated data are delivered to the sink node and published there. In this chapter, the implementations of these two phases are studied, in terms of energy efficiency and bandwidth consumption.

In this chapter, the physical signal *field* is utilized for in-network processing. In fact, any WSN implementation aims to monitor some physical incidents, such as flood, wild fire, pollution and traffic. In the area that a WSN is monitoring, there must be a physical signal. Additionally, the intensity (strength) of this physical signal varies spatially. Within the area, the intensity of the physical signal forms a *field*. At a source node, the generated sensory data should correspond to the intensity of the physical signal. Thus, with the sensory data from all source nodes collected, the physical signal *field* can be figured out. Along the gradient direction of the *field*, sensory data can then be converged to some local nodes.

In Section 6.2, the signal *field* surrounding a physical incident is analyzed in detail. In section 6.3, the designed algorithms are provided. In section 6.4, the performance of the design is analyzed through simulations.

6.2 Analysis

For a physical object that a WSN is monitoring, there must be a physical signal emitted from the object. The physical signal can be sensed by the sensor nodes of the WSN. Moreover, the physical signal forms a *field*. This can be illustrated by the following example.

Example: a group of people spread in an area and act as sensor nodes. Now, a bear comes. Somebody may see that the bear is coming; somebody may see that a black dot is moving; and somebody may see nothing. When the bear goes closer to a person, the people can see the bear clearer. When a person is standing too far away from the bear, the person doesn't know at all that the bear is around.

The example above can be described in another way: there is a sight *field* surrounding the bear. At each *location*, the *field* intensity is related to the distance between the *location* and the bear. If a *location* is closer to the bear, the *field* intensity at this *location* is higher; at this *location*, the person may see the bear clearer. If another *location* is far away from the bear, the *field* intensity of this *location* goes to zero; at this *location*, the person can not see the bear.

Therefore, it is the physical signal *field* that a WSN can sense. At each *location*, the intensity of the *field* corresponds to the amplitude of the sensed signal. Moreover, the amplitude of the sensed signal corresponds to the value of sensory data. The physical signal *field* bridges the physical object to the sensory data. In addition, the temporal variation of the sensory data indicates that a physical incident occurs

Physical signal *field*s of different objects can overlap with each other. In a three-dimensional space, if the height corresponds to the intensity of physical signal *field*s, the shape of physical signal *field*s may look like mountains spanning across the area.

In this research, the physical signal *field* is explored to cluster sensor nodes. As shown in Figure 6-1, the area can be divided into small pitches by valleys and saddles. This division can also be leveraged across node clustering. As a result, there is no overhead cost for node clustering. Moreover, this clustering scheme adapts to the spatial variation of the *field*, especially for the cases when some sensor nodes die.

In this research, the physical signal *field* is also explored for routing. In fact, *field*-based routing is not new. In [Poo00], packets can be driven along the gradient direction of the cost *field*. In this chapter, the physical signal *field* is a substitute for the cost *field*. Along the gradient direction of the physical signal *field*, sensory data are converged to *mountain peak*s of their local clusters. At the *mountain peak*s, sensory data are aggregated. In this way, in-network processing is conducted.

6.3 Algorithm

Within WSNs, the routing scheme should be divided into two phases. The first phase works for in-network processing. The second phase works for delivering packets to the sink node. Corresponding to these two phases, there are two tags. One tag is "sink routing" (SR); the other tag is "in-network processing" (INP). If a delivered packet is in the first phase, it should be assigned by INP; if a delivered packet is in the second phase, it should assign by SR.

It is assumed that the scheme of sink routing is given. (The sink routing can be Reverse Path Forwarding, Cost value field Forwarding, Geographical Forwarding). In this section, the routing schemes for the first phase (i.e. in-network processing) are presented.

Gradient directions of a physical signal field can be used for in-network processing in many ways. For example, gradient directions can be used directly, and can drive sensory data to local peaks of a physical signal field. Also, gradient directions of two different fields can be mixed together to drive the sensory data's deliveries. In this design, two routing schemes are proposed for in-network processing. The first routing scheme is called Center Convergence Scheme (CCS), which routes sensory data to local peaks of the physical signal field. The second routing scheme is called Linear Combination Scheme (LCS), in which the routing direction of each hop is derived from a linear superposition of two gradient directions. These two gradient directions are derived from two different fields: one gradient direction is derived from the physical signal field, and the other gradient direction is from the cost value field initiated by the sink node.

6.3.1 Center Convergence Scheme (CCS)

In CCS, gradient directions of a physical signal field are used for in-network processing. Whenever a message (data packet) is generated at a node, it will be tagged by INP first. After that, the node probes among its neighbor nodes to find out which node senses the strongest physical signal. This is called the probing process. If the signal intensity at the node is higher than that at all its neighbor nodes, the tag of the message should change to SR. Otherwise, the message should be sent to the neighbor node with the highest signal intensity.

Similarly, if a message is received from some other node, its tag should be checked first. For the message with SR, it will be delivered according to the given sink routing scheme. For the message with INP, a probing process will be invoked. In addition, a memory buffer is defined at each node for the message (data) aggregation. CCS consists of two programs that need to be run separately and simultaneously at each node. Those two programs are written in pseudo codes that follows the structure of C language.

Table 6-1. Buffer control of CCS scheme

```
If (message is generated at the node)
{   Add the tag INP to the message;
    Put the message into the buffer;
} else if (message is received from other nodes)
{   check the tag of the message;
    If (the tag is INP)
    {   put the message into the buffer;
    } else if (the tag is SR)
    {   deliver the message to sink;
    }
}
```

Table 6-2. Publishing control of CCS scheme

```
while (the pre-defined timer runs out)
{  aggregate messages in the buffer;
   Let Val = data sensed by the node;
   Probe Val of neighbor nodes;
   If (exist neighbor with greater Val)
   {   title the aggregated message with INP;
       Select the neighbor with greatest Val;
       Send the message to this neighbor;
   } else
   {   title the aggregated message with SR;
       Deliver the message to sink;
   } empty the buffer;
   Reset the time;
}
```

The first program checks the tag of messages. When a message (data packet) arrives, its tag will be checked first. If the tag is INP, then the message will be put into the memory buffer; if not, its tag must be SR, and the message will be delivered to the sink by following the sink routing scheme. It is assumed that the scheme of sink routing is given. If a message is generated at the node, the message will always be tagged INP initially and put into the buffer. The pseudo codes are shown in Table 6-1.

The second program has two tasks: one is to probe the node's neighbors in order to find the local peak of the physical signal field and the other task is to aggregate messages. When a predefined timer runs out, all messages in the memory buffer are aggregated. After that, a tag is assigned to the aggregated message. If the node is at a local peak of the physical signal field, the aggregated message should be tagged "SR" and delivered to the sink node under the given sink routing scheme. Otherwise, if the node is not at a local peak of the physical signal field, the aggregated message is tagged as "INP" and forwarded along the gradient direction of the physical signal field.

6.3.2 Linear Combination Scheme

The LCS design aims at in-network processing. In addition, LCS requires that the sink routing scheme should be "Cost value field Based Forwarding" (GRAB).

At this point, there are two fields over a WSN. They are the cost value field and the physical signal field. At each node or at each point where a node locates, two gradient directions exist corresponding to those two fields. In the design of LCS, the two gradient directions are linearly combined. The combined direction is used for driving sensory data convergence (in-network processing).

For the cost value field, the minimum cost value "0" is at the sink node and the whole field looks like a funnel when we view the cost value as the height of a three-dimensional space. For the physical signal field, it looks like a mountain (which is an overturned funnel), when the field intensity is viewed as the height of the three-dimensional space. In this design, all cost values of the cost field are multiplied by −1 in order to turn the "funnel" over. At each node, let its cost value be V_c and its sensed signal intensity value of the physical signal field be V_i, then its linear superposition value ($V_{\text{L.S.}}$) is $V_{\text{L.S.}} = -\alpha V_c + \beta V_i$, ($\alpha$ and β are constant factors larger than zero).

Table 6-3. Buffer control of LCS scheme

```
If (message is generated at the node)
{     Add the tag INP to the message;
      Put the message into the buffer;
} else if (message is received from other nodes)
{     check the tag of the message;
      If (the tag is INP)
      {     put the message into the buffer;
      } else if (the tag is SR)
      {     deliver the message to sink;
      }
}
```

When a message is generated at a node, the message is tagged by "INP" first. (All messages should be tagged either by "INP" or by "SR.") For any message with an SR tag, the node will deliver it to the sink when the routing scheme of GRAB (cost value field routing) is followed. For the message with an INP tag, a probing process would be invoked at the node. The probing process is almost the same as that of the central convergence scheme (CCS), except the probing value. Here, the probing value (which is the value of signal intensity in CCS) is changed to be the linear super-position value ($V_{L.S.}$). If the $V_{L.S.}$ of the node is the highest, then the tag of the message will become "SR"; otherwise, the message will keep its "INP" tag and will be forwarded to the node's neighbor with the highest $V_{L.S.}$.

Table 6-4. Publishing control of LCS scheme

```
while (the timer is run out )
{   aggregate all messages in the buffer;
    $V_{LS} = -\alpha Vc + \beta Vi$;
    probe $V_{LS}$ of neighbor nodes;
    if (exist a neighbor with larger $V_{LS}$)
    {     tag the aggregated messages with INP;
          select the neighbor with largest $V_{LS}$;
          send the aggregated message to;
    } else
    {     tag the aggregated messages with SR;
          send the aggregated messages;
          toward sink by cost value field routing;
    }empty the buffer;
    reset the timer;
}
```

Two programs are listed here. These two programs should be run at each node separately and simultaneously. Inside each node, these two programs exchange data through the memory buffer of the node. For the first program, if a message has an INP tag, then it will be put into the memory buffer. If a message's tag is SR, then the message will be delivered to the sink node by following the routing scheme of GRAB. Messages generated at the node will always be tagged with "INP" first, and then put into the memory buffer. The first program is the same as that of the center convergence scheme (CCS), and shown in Table 6-3.

The second program is in charge of probation and aggregation. In addition, a timer is pre-defined to synchronize the entire process. When the timer expires, all messages in the memory buffer are aggregated. After that, a probing process is invoked. If the node possesses the highest $V_{\text{L.S.}}$ among all its neighbor nodes, the aggregated message is tagged with "SR," and delivered by following the scheme of cost value field forwarding (GRAB). Otherwise, the aggregated message is tagged INP and sent to the neighbor node with the highest $V_{\text{L.S.}}$.

6.3.3 Analysis of CCS and LCS

If the sink routing scheme is set to be Cost value field Based Forwarding, CCS is only a special case of LCS when $\alpha = 0$. In this sub-section, the performance of LCS will be discussed with α and β adjusted, and the performance of LCS and CCS will be compared.

To simplify the process of analysis, it is assumed that the physical signal field covers a round area with a radius R. The highest intensity point (peak) of the field is at the centre of the round area. As shown in Figure 6-2, the intensity distribution is:

$$\begin{aligned} V_i &= 0, && \text{for } \rho > R; \\ V_i &= c_1/\rho^2, && \text{for } R \geq \rho \geq \varepsilon; \\ V_i &= c_1/\varepsilon^2, && \text{for } \varepsilon > \rho > 0. \end{aligned} \tag{6-E$_1$}$$

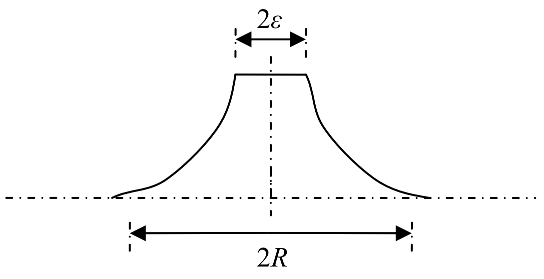

Figure 6-1. The distribution of physical signal field

Here, ρ is the distance to the field center (peak). $c_1 > 0$ and $\varepsilon > 0$ are two constants.

As shown in Figure 6-3, the distance from the sink node to the centre of the physical signal field is D ($D > R$). For each node, its cost value is $V_c = c_2d$, in which d is the distance to the sink, and c_2 is another constant (c_2>0).

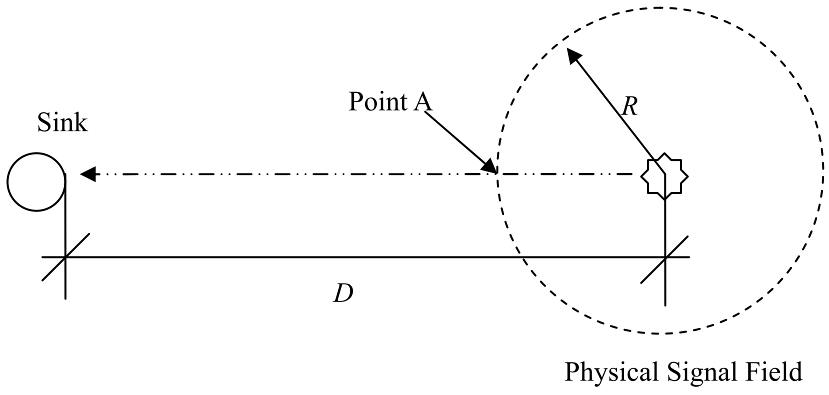

Figure 6-2. Positions of sink and physical signal field

After the linear superposition ($V_{\mathrm{L.S.}} = -\alpha V_c + \beta V_i$), the original physical signal field changes to a new field. In addition, the original area of the physical signal field area is folded and separated into two parts with two peaks, respectively, as shown in Figure 6-4.

It is clear that the maximum value of $V_{\text{L.S.}}$ should be on the straight line (Figure 6-3) that connects the sink node and the center of the physical signal field. Along the straight line, let ρ be the distance to the center of the physical signal field. This produces,

$$V_{\text{L.S.}} = -\alpha c_2(D - \rho) + \beta c_1/\rho^2 \qquad (6\text{-E}_2)$$

Let the derivative of LSV be zero, and it yields

$(V_{\text{L.S.}})' = \alpha c_2 - 2\beta c_1/\rho^3 = 0$, and,

$$\rho^3 = 2\beta c_1/(\alpha c_2) \qquad (6\text{-E}_3)$$

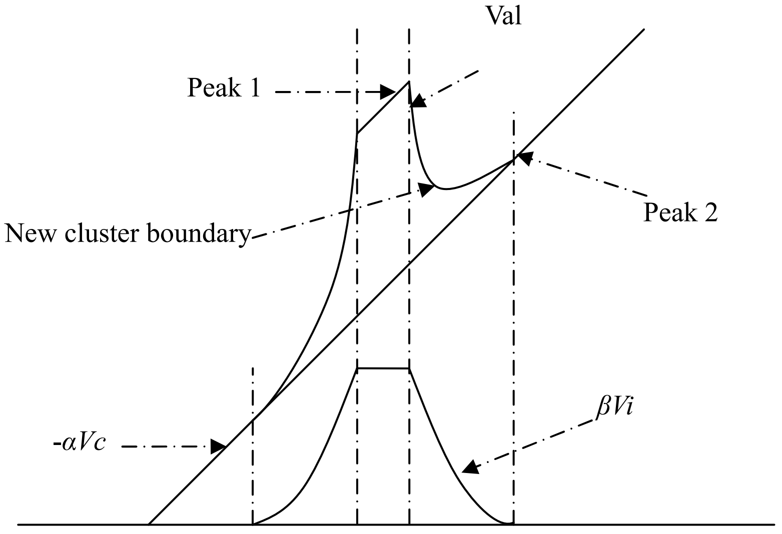

Figure 6-3. The field after LCS scheme

Therefore, the new cluster boundary, (or the minimum value of the maximum $V_{\text{L.S.}}$s,) is at $\rho = [2\beta c_1/(\alpha c_2)]^{1/3}$. Furthermore, when several adjacent physical signal fields exist, this new cluster boundary introduces a new clustering scheme which is shown in Figure 6-4. From the figure, it can be observed that the local peaks (that sensory data converge to) are no longer at the centers of the clustered area, but shift toward the sink node.

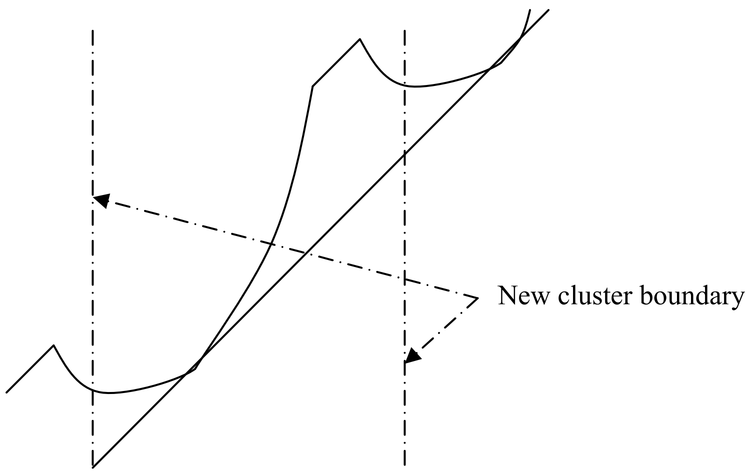

Figure 6-4. New cluster boundary for adjacent signal fields

Next, it is necessary to analyze the communication expense of in-network processing, and the variation of the communication expense, which is caused by the local peaks' shift. Two cases are taken into account. For the first case, the convergence point is at the center of the physical signal field. For the second case, the convergence point is at point A of Figure 6-5. In order to clarify the analysis, it is assumed that all nodes are distributed among the area continuously and transmission conflicts are ignored. Within the physical signal field, each node should generate a message. The communication expense (energy expense) of a message delivery is equal to the distance that the message travels. In addition, all messages would not be aggregated along its way. All messages are aggregated at the convergence point.

When the convergence point is at the center of the physical signal field area

The total communication expense is $\int_0^R \rho 2\pi\rho \, \mathrm{d}\rho = (2/3)\pi R^3$ (6-E_4)

When the convergence point is at point A, as shown in Figure 6-5,

The total communication expense is $\int_0^R \rho 2\arccos\left(\rho / 2\rho\right) \rho \, d\rho = 80R^3/9$ (6-E$_5$)

From the results above, when the convergence point is at point A, the total communication expense of the data convergence is about four times greater than the total communication expense when data converge to the center of the physical signal field. However, the distance from point A to the sink is shorter than the distance from the center of the physical signal field to the sink. Thus, there is a trade-off with respect to selecting the convergence point between point A and the center of the physical signal field.

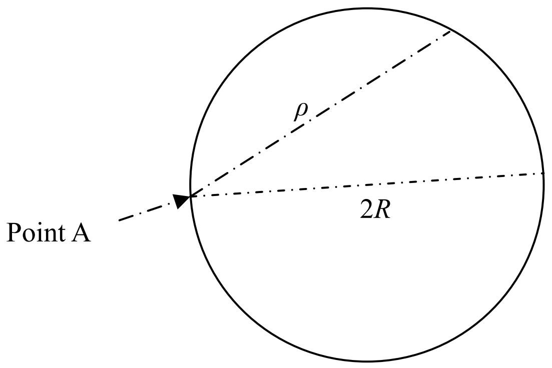

Figure 6-5. Illustration the meaning of variables ρ and R

By the equation (6-E$_3$) $\rho^3 = 2\beta c_1/(\alpha c_2)$, and the range limits of ρ ($\varepsilon \le \rho \le R$), the range limits of β/α can also be ascertained; that is $c_2\varepsilon^3/(2c_1) \le \beta/\alpha \le c_2R^3/(2c_1)$. If $\beta/\alpha > c_2R^3/(2c_1)$, messages are converged to the peak of the physical signal field, and the cluster boundary is the same as the boundary of the physical signal field (the same as CCS). If the physical signal field is isolated from the others, and $\beta/\alpha < c_2\varepsilon^3/(2c_1)$, messages from the physical signal field area are converged to "point A". If several physical signal fields are adjacent to each other and $\beta/\alpha < c_2\varepsilon^3/(2c_1)$, all those physical fields are soldered into a big field, and messages are converged to “point A” of this new big field. If several physical signal fields are adjacent to each other, and $c_2\varepsilon^3/(2c_1) \leq \beta/\alpha \leq c_2R^3/(2c_1)$, nodes will be clustered by the new boundary as shown in Figure 6-5, rather than by the natural boundary of the physical signal fields.

6.4 Simulation

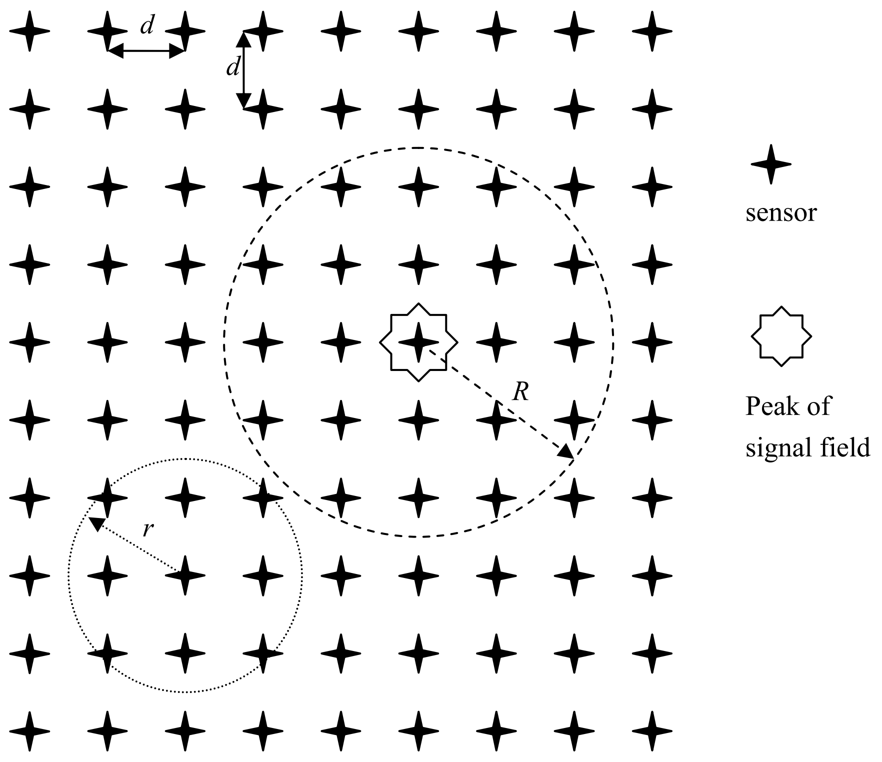

Figure 6-6. Simulation model

Compared to other schemes, this design can cluster nodes with zero overhead. In addition, the routing paths within a cluster are constructed automatically. Moreover, the scheme will be robust, even if some nodes die.

Simulations run over a square area ($400\times400\text{m}^2$). Within the area, nodes are settled at cross-points of a uniform square grid, with the grid spacing *d*. *d* is selected to be 12m, 16m or 20m, respectively. The transmission radius of each node is *r*, which is set to be 25m. It is clear that *r* is greater than *d*. A physical signal field covers a round area with radius *R*. During the simulation, *R* varies from 10m to 100m. The peak of the field is at the center. Only the communication expense of in-network processing is considered.

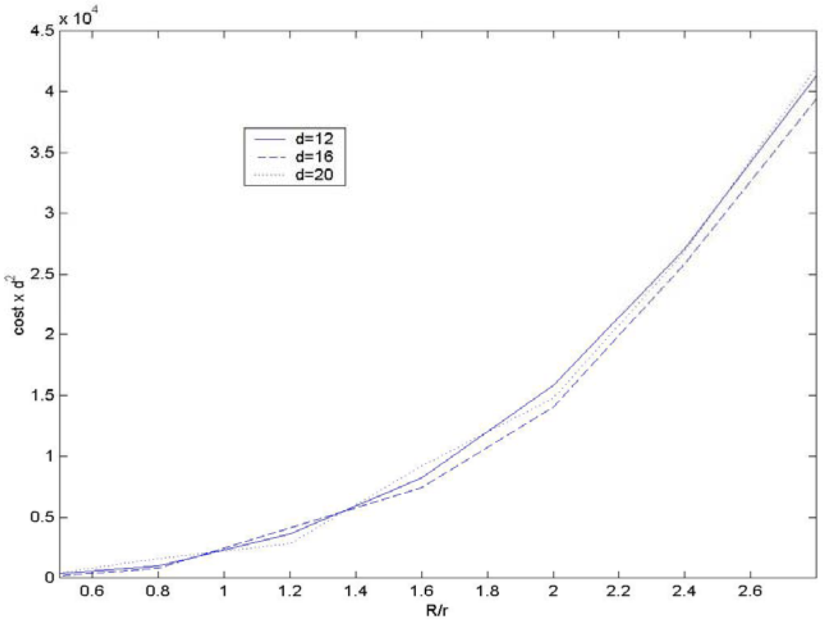

Figure 6-7. In-network processing expense per node

As shown in Figure 6-8, three curves (which correspond to the cases of d = 12, 16, and 20, respectively) overlap each other. Since the node density of the network is $1/d^2$, this also implies that the expense is in direct proportion to the node density. In addition, the total communication expense goes up sharply when the ratio of *R*/*r* increases. For a given physical incident, the radius of its physical signal field *R* is fixed, and only *r* can be varied. Thus, by increasing *r*, the expense of in-network processing can be cut down sharply. However, by increasing *r*, more nodes will be covered inside a broadcast area. This may incur more transmission collision. As shown in Figure 6-9, the average communication expense (cost/R^2) goes up with the increasing R/r and node density ($1/d^2$).

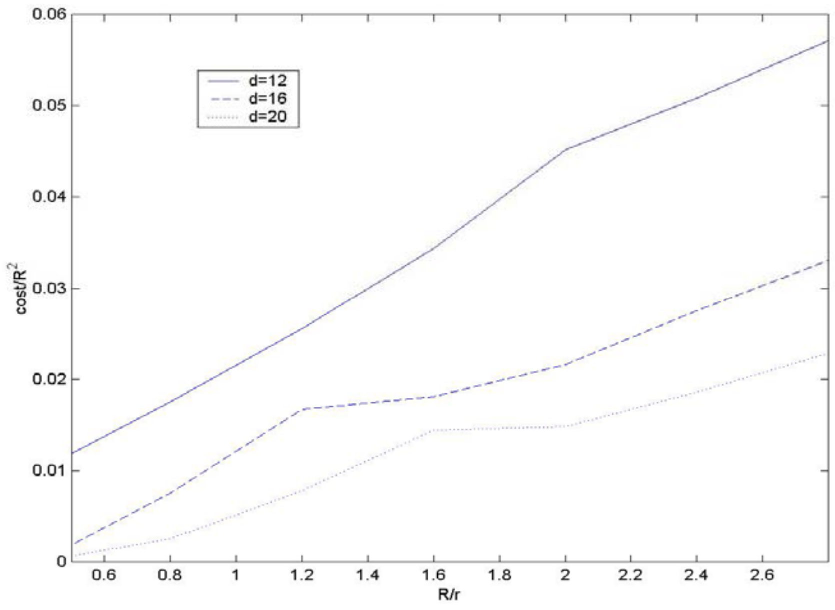

Figure 6-8. In-network processing cost per area

To investigate the performance of LCS when β/α is varied, let *R* be 50m. In addition, let the convergence point move from the center of the physical signal field to the boundary of the field to imitate the variation of β/α. Here, “offset” denotes the distance that the convergence point deviates from the center of the field. As shown in Figure 6-10, all three curves are close to each other, which is similar to cases shown in Figure 6-8. Roughly, with the increasing offset (or the convergence point moving toward the boundary of the physical signal field), the average communication expense of in-network processing (cost×d^2) goes up.

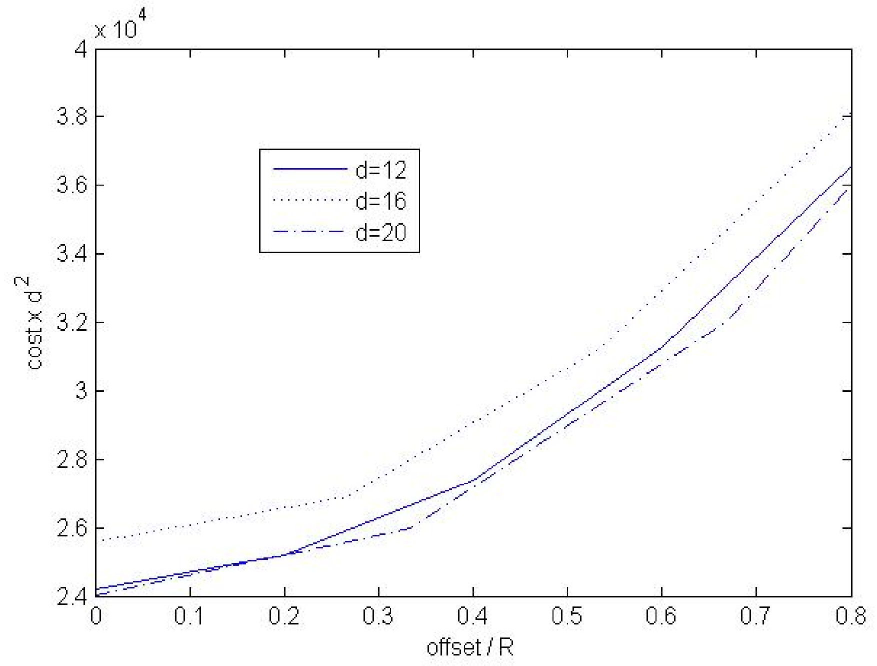

Figure 6-9. Effect of offset on in-network processing cost per node

Frequently, sensor nodes may run out of their battery and die, and holes will emerge inside the WSN area, as shown in Figure 6-11. For a hole with a convex boundary, the designs could still make data packets detour the hole for in-network processing. If some holes are with concave boundaries, data packets can not detour the holes by following the routing designs. In such cases, the clustering designs will split the physical signal field area into two small clusters, and data packets in the area will converge to their own local peak of these two small clusters, respectively. In Figure 6-11, in addition to the peak of the physical signal field, the *location* of the sensor node P will act as a new local peak.

All nodes inside the physical signal field are split and grouped into two clusters. In fact, concave holes are common in WSNs, and can prevent a cluster from being too big.

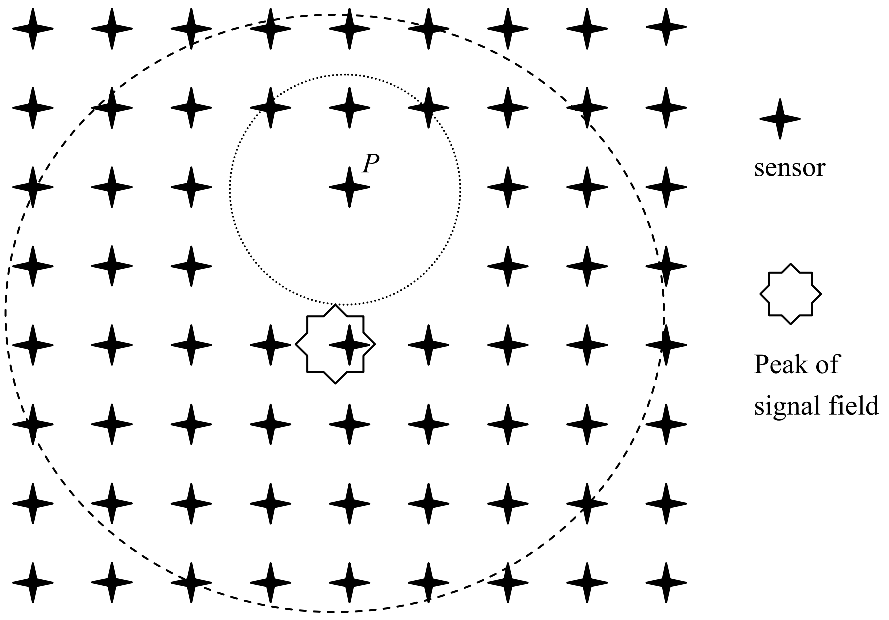

Figure 6-10. Concave hole in wireless sensor network

Summary

In the chapter, the technology of “in-network processing” is extended, with regards to physical incident publishing. Here, physical signal fields are explored. According to the designs, sensory data are converged to some local nodes and aggregated there, along with the gradient direction of the physical signal fields. In this way, the overhead cost of “in-network processing” is reduced, as well as the cost of physical incident publishing. Based on the study of Chapters 4, 5, and 6, the incident publishing has been inspected thoroughly. In the next chapter, the main results of this research would be reviewed.

Chapter 7 Conclusion

The contributions of this research can be categorized into two aspects: 1). the study and designs for logging incidents; 2). the study and designs for publishing incidents. Since the two aspects (i.e. designs, implementation and optimization of incident logging and publishing) are fundamental for network forensic support, this dissertation is expected to lay a cornerstone in the forensic research concerning mobile networks.

The first two sections of this chapter summarize the main results: Section 7.1 presents contributions related to incident logging; and section 7.2 presents contributions related to incident publishing. In the last section, Section 7.3, the future research is considered.

7.1 Logging Network Incidents

In addition to *time stamp*, *location* is another key component of an incident record, because *location* indicates the place where the incident occurs. For network topology spaces, the concept of *distance metric* is not available. As a result, the *location* of a node can not be measured without a *distance metric* given. In this research, an innovative concept (i.e. Background) is proposed to label *location*s of nodes. According to this research, Background of a node (defined as the neighbor list of the node) is used as a substitute for the node's *location*. Moreover, the Background of a node at a particular time can be viewed as private information of the node, and can even be used to check the identity of the node.

In Chapter 3, the Background and its applications are studied, and a design has been extended to authenticate ad hoc nodes. According to the design, participating nodes of a MANET should broadcast their neighbor lists periodically. Correspondingly, each node would receive the broadcasted packets from its neighbors, and update its *background table* based on the packets (carrying neighbor lists of its neighbors). In addition, each node should log its *background table* after a given time period.

Based on these logged background tables, the process of node authentication (i.e. Background Checking) can then be conducted. At the beginning of the process, a node would be requested to present its Background at a random time point. From the response (the node's Background at this time point) of the node, several neighbors of the node at this time point are selected as references of the node. After that, the references of the node are contacted, in order to check if the response from the node is conformed to the logs of its references.

The effectiveness of the authentication design has been tested by the logical analysis. Furthermore, simulations examine the side-effects of the performance for this authentication design. The side-effects include bandwidth consumption and delivery delay. The results from the simulations show that the overhead cost of this authentication design does not impair the performance of mobile ad hoc networks.

7.2 Publishing Network Incidents

"Incident logging" makes network forensics possible, and "incident publishing" makes network forensics feasible. Log files are the resources on which network forensic analyses should be based. However, for tracking a particular network incident, how to find the related log files is still a question.

In the designs of Chapter 4, distributed log files are published and organized to facilitate the process of incident tracking. In order to efficiently find all related information of an incident from numerous log files, these log files should be classified and sorted beforehand. Moreover, the design of the classifications and sorting processes must be effective in dynamic network circumstances in which log files reside on mobile nodes. More importantly, log files must be published willingly in order to have them organized.

This research contributes a new framework of organizing logs by extending the P2P technology of Distributed Hash Tables (DHTs). Furthermore, traditional designs regarding network traceability (i.e. SPIE, PPM, and ITrace) are incorporated in this new framework. Under the architecture of a DHT design, log files are hashed and published on some pre-selected nodes, which are determined by the DHT design.

However, there is a recursion problem during the processes of log publishing. Since the delivery of a publishing packet is also a network incident, it needs to be logged at intermediate nodes along the packet's delivery path. Furthermore, logs incurred by the delivery of publishing packets also need to be published. In this way, publishing processes would be invoked recursively. As a result, even if one logged incident is trying to publish, the processes of log publishing would be invoked endlessly. In the end, these recursively populated publishing packets would consume the entire bandwidth of the network. In fact, this recursive issue is common in any design of log publishing, including the DHT-based designs.

In this research, a new scheme, Packet-Defined Probabilistic Delivery (PDPD or PD2), is developed to overcome this recursion problem. According to the scheme PD2, each packet is assigned with a value (i.e. delivery rate) and should be forwarded with the probability of this value. In addition, a function $f(\cdot)$ is shared by all participating nodes, in order to control this value (i.e. delivery rate). With the PD2 design, the total number of populated publishing packets is limited throughout the recursive processes. Details of the function $f(\cdot)$ are also discussed in order to optimize the performance of PD2. Rules and criteria for the function construction are listed, as are some instances of $f(\cdot)$. At the end of the chapter, the performance of the proposed scheme PD2 is analyzed, and its effectiveness is examined by simulations.

In Chapter 5, log (or incident) publishing is discussed in GSM networks, and is specified as *location* publishing. In addition, a specific forensic problem, Movement Tracking, is considered. According to the Standard IS-41 (a GSM standard), mobile subscribers of a GSM network should always have their current *location*s published (or registered) on their HLRs. As a result, the subscriber's movement can be traced by referencing their HLRs. In this research, technologies of *location* publishing are widely explored. Based on the design of "anchor chain" [Bej03], a new design has been also developed. This new design follows the structure of "anchor chain," but modifies the criterion of "chain update," in order to improve the performance. The theoretical analysis shows that the design can sharply reduce the accumulative cost of "*location* publishing". In addition, the efficiency of the design is examined by simulations. More importantly, the new design would track the subscriber's movement fast.

In Chapter 6, the incident publishing is studied in the context of Wireless Sensor Networks, in terms of "physical incident publishing." An innovative design has been presented, in order to efficiently publish sensory data (or physical incidents). According to the design, the physical signal fields, which sensor nodes are monitoring, are adopted to direct local data convergence. This implies that not only artificial architectures (such as GSM or DHTs), but also natural architectures (such as physical signal fields) can be used for incident/log publishing. At the end of the chapter, the robustness and reliability of the design are examined by simulations.

7.3 Future Work

7.3.1 Refinement and Extension

This dissertation suggests that many research issues need to be further explored, such as selecting hash functions and particular DHT structures, constructing of the reflexive decreasing function $f(p)$, etc.. In addition, the improvement of “anchor chain” design, which is developed in GSM networks, should be implanted into ad hoc networks in order to widen Movement tracking. For this implantation, problems still exist as to how to select anchors from mobile nodes and how to update these anchors. Likewise, the designs of in-network processing have similar problems when being implanted into mobile environment. Therefore, future research needs to focus on tackling these problems. Moreover, the modern techniques of “intrusion detection” should be applied to the processes of “logging incidents” and “publishing incidents,” in order to classify incidents with different suspicious levels.

The *background table* of each node contains information of the local topography surrounding the node. By logging the *background table* along with time stamps, the node has actually recorded the variation of its local topography. Moreover, the global topography of the entire network can be constructed by collecting the logged *background table*s of all nodes. As the global topography must be helpful in network forensics, future work can be conducted, in response to constructing the global topography from the distributed *background table*s.

7.3.2 Issues beyond the Dissertation

So far, discussions and analyses are surrounding the fundamental issues on which forensic activities should base. However, it is also necessary to explore some network services with regards to distributing the outcomes of a forensic analysis. For example, the forensic analyst identifies a mobile node as malicious. The analyst also needs to inform all other nodes so that they can isolate this malicious node. Therefore, a network service of distributing the analysis results is necessary. Meanwhile, this service should avoid directly distributing the results to this malicious node. Furthermore, whenever a node finds that the malicious node is wandering, an alarm should be activated.

Also, all network services, designed or expected to be designed, aim to support a general forensic analysis, instead of tracking a specific attack. In fact, those network services may be designed more delicately if they focus on some specific attacks or misbehaviors. For example, for SYN flooding attacks, these attacking packets consist of the same destination address and contents, but different source addresses and packet sequence numbers. At each mobile node, a forensic mechanism can be designed in order to identify the SYN flooding attacks. Whenever a mobile node receives three packets with the pattern of a SYN flooding attack (i.e. with the same destination address and contents, but different source addresses and packet sequence numbers,) in a short duration (say three seconds), the node should refuse to forward these packets for a while (say five minutes). Here, time periods (three seconds and five minutes) can be adjusted via optimization.

Overall, there are two sub-fields in the research of network forensic support: 1) network services used before forensic analyses, and 2) network services used after forensic analyses. Within each sub-field, designed services may focus on a general purpose or on some specific attacks. This research only studies some network services used before general forensic analyses, and leaves the other untouched. Thus, future research will focus on these untouched issues, with the goal of fulfilling the needs of network forensics.

LIST OF REFERENCES

[Aad04]. I. Aad, J.P. Hubaux, and E. Knightly, “Denial of Service Resilience in Ad Hoc Networks”, in *Proceedings of the 10th annual international conference on Mobile computing and networking (MobiCom’04)*, pp. 202-215, 2004.

[Bea03a]. J. Beaver, M. A. Sharaf, A. Labrinidis, and P. K. Chrysanthis, “Location-Aware Routing for Data Aggregation in Sensor Networks”, in *Proceedings of Geo Sensor Networks Workshop*, Portland, Maine, Oct 9-11, 2003.

[Bea03b] J. Beaver, M. A. Sharaf, A. Labrinidis, and P. K. Chrysanthis, “Power-Aware In-Network Query Processing for Sensor Data”, in *Proceedings of the 3rd ACM MobiDE Workshop*, San Diego, CA, 2003.

[Bej03] Y. Bejerano, I. Cidon, “An Anchor Chain Scheme for IP Mobility Management”, *Wireless Networks*, vol. 9(5), pp. 409-420, 2003.

[Bel03]: S. M. Bellovin, M. Leech, and T. Taylor, “ICMP traceback messages”, in Internet Draft draft-ietf-itrace-04.txt (Work in progress), IETF, Feb 2003.

[Bis05] R. Biswas, K. Chowdhury, and D. P. Agrawal, “Attribute Allocation in Large Scale Sensor Networks”, *in Proceedings of the 2nd international workshop on data management for sensor networks*, pp. 27-33, 2005.

[Bre00] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, and H. Yu, “Advances in network simulation”, *IEEE Computer*, vol. 33(5), pp. 59-67, May 2000.

[Cap03] S Capkun, L. Buttyan, and J. P. Hubaux, “Self-organized public-key management for mobile ad-hoc networks”, *IEEE Trans. on Mobile Computing*, vol. 2(1), pp. 52-64, 2003.

[Cap06] S. Capkun, J. Hubaux, and L. Buttyan, “Mobility Helps Peer-to-Peer Security”, *IEEE Trans. on Mobile Computing*, vol. 5(1), pp. 43-51, 2006.

[Cha05] I. Chatzigiannakis, S. Nikoletseas, and P. Spirakis, “Efficient and Robust Protocols for Local Detection and Propagation in Smart Dust Networks”, *Mobile Networks and Applications*, vol. 10(1), pp. 133–149, 2005.

[Dea02]: D. Dean, M. Franklin, and A. Stubblefield, “An algebraic approach to IP traceback”, *ACM Trans. on Information and System Security (TISSEC)*, vol. 5(2), pp. 119-137, May 2002.

[Fan03] Q. Fang, F. Zhao, and L. Guibas, “Lightweight Sensing and Communication Protocols for Target Enumeration and Aggregation”, in *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc’03)*, pp. 165-176, 2003.

[Gan04] L. Gan, J. Liu, and X. Jin, “Agent-Based Energy Efficient Routing in Sensor Networks”, in *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS’04)*, vol. 1, pp. 472-479, 2004.

[Go05] H. W. Go, P. Y. Chan, Y. Dong, et al., “Performance evaluation on CRL distribution using flooding in mobile ad hoc networks”, in *Proceedings of the 43rd annual Southeast regional conference*, Session: Networking and mobile computing, vol. 2, pp. 75 – 80, 2005.

[Gon05]: C. Gong, K. Sarac, “IP Traceback Based on Packet Marking and Logging”, in *Proceedings of the IEEE International Conference on Communication (ICC’05)*, pp. 16-20, 2005.

[Hai01] H. Safa, S. Pierre, and J. Conan, “An efficient location management scheme for PCS networks”, *Computer Communications*, vol. 24(14), pp. 1355-1369, 2001.

[Hai02] H. Safa, S. Pierre, and J. Conan “A built-in memory model for reducing location update expense in mobile wireless networks” *Computer Communications*, vol. 25(14), pp. 1343-1353, 2002.

[Hei01] J. Heidemann, F. Silva, and C. Intanagonwiwat, “Building Efficient Wireless Sensor Networks with Low-Level Naming” in *Proceedings of the 18th ACM symposium on Operating systems principles*, Session: Networking, pp. 146-159, 2001.

[Ho95] J. S.M. Ho, I. F. Akyildiz, “Local Anchor Scheme for Reducing Location Tracking Expense in PCNs”, in *Proceedings of the 1st annual international conference on Mobile computing and networking (MOBICOM’95)*, pp. 181-193, 1995.

[Hon04] B. Hong, V. K. Prasanna, "Optimizing a Class of In-network Processing Applications in Networked Sensor Systems", in *Proceedings of the 1st IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS'04)*, Session: Coding/Compression, pp. 154-163, 2004.

[Hua03] H. Huang, S. F. Wu, "An approach to certificate path discovery in mobile Ad Hoc networks", in *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, Session: Authentication, pp 41-52, 2003.

[Hua05]: Y. Huang, W. Lee, "Hotspot-based traceback for mobile ad hoc networks", In *Proceedings of the 4th ACM workshop on Wireless security (WiSe'05)*, Session: Misbehavior, pp. 43-54, 2005.

[Hub01] J.-E Hubaux, L. Buttyin, and S. Capkun, "The quest for security in mobile ad hoc networks", In *Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'01)*, Session: Security, pp. 146-155, 2001.

[Hus06]: A. Hussain, J. Heidemann, and C. Papadopoulos, "Identification of Repeated Denial of Service Attacks", in *Proceedings 25th IEEE International Conference on Computer Communications (INFOCOM'06)*, pp. 1-15, 2006.

[Kal99] A. Kaloxylos, E. Zervas, and L. Merakos, "Location Management in Wireless ATM Customer Premises Networks", Technical Report TR98-0008, University of Athens (Greece), 1998.

[Kon01] J. Kong, E Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing robust and ubiquitous security support for MANET", In *Proceedings of the 9th IEEE International Conference on Network Protocols (ICNP'01)*, pp. 251-260, 2001.

[Kol99] A. N. Kolmogorov, S. V. Fomin, "Elements of the Theory of Functions and Functional analysis", Dover Publications, 1999, ISBN: 978-0486406831.

[Kri02] B. Krishnamachari, D. Estrin, and Stephen Wicker, "The Impact of Data Aggregation in Wireless Sensor Networks", in *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDC'02)*, pp. 575-578, 2002.

[Kuz03]: A. Kuzmanovic, E. Knightly, “Low-Rate TCP-Targeted Denial of Service Attacks (The Shrew vs. the Mice and Elephants)”, in *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM’2003)*, Session: Deny-of-service, pp. 75-86, 2003.

[Lee01] S. Lee, C. Shields, “Tracing the source of network attack: A technical, legal and societal problem”, In *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security*, pp. 239-246, 2001.

[Li04] J. Li, Y. Pan, and Y. Xiao, “A Dynamic HLR Location Management Scheme for PCS Networks”, in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM’2004)*, vol. 1, pp. 266-276, 2004.

[Ma04] W. Ma, Y. Fang, “Dynamic Hierarchical Mobility Management Strategy for Mobile IP Networks”, *IEEE Journal on selected area in Communications*, vol. 22(4), pp. 664-676, 2004.

[Mao04] Z. Mao, C. Douligeris, “A distributed database architecture for global roaming in next-generation mobile networks”, *IEEE/ACM Trans. on Networking*, vol. 12(1), pp. 146-160, 2004.

[May02] P. Maymounkov, D. Mazieres, “Kademlia: A peer-to-peer information system based on the XOR metric”, in *Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS’02)*, pp. 53-65, 2002.

[Mer05] J. van der Merwe, D. Dawoud, and S. McDonald, “Fully Self-Organized Peer-to-Peer Key Management for Mobile Ad Hoc Networks”, in *Proceedings of ACM Workshop on Wireless Security (WiSe'05)*, Session: Key management, pp. 21-30, 2005.

[Mer07] J. van der Merwe, D. Dawoud, and S. McDonald, “A Survey on Peer-to-Peer Key Management for Mobile Ad Hoc Networks”, *ACM Computing Surveys (CSUR)*, vol. 39(1), pp. 1-45, 2007.

[Mit02]: A. Mitchell, and G. Vigna “MNEMOSYNE: Designing and implementing network short-term memory”. in *Proceedings of the 8th IEEE International Conference on Engineering of Complex Computer Systems*, pp. 91-100, 2002.

[Par01]: K. Park, H. Lee, “On the effectiveness of route-based packet filtering for distributed DoS attack princidention in power-law internets”. *ACM SIGCOMM Computer Communication Review*, vol. 31(4), pp. 15-26, 2001.

[Pax01]: V. Paxson, “An analysis of using reflectors for distributed denial-of-service attacks” *ACM SIGCOMM Computer Communication Review*, vol. 31(3), pp. 1-10, 2001.

[Pie06] R. D. Pietro, L. V. Mancini, and A. Mei, “Energy efficient node-to-node authentication and communication confidentiality in wireless sensor networks”, *Wireless Networks*, vol. 12(6), pp. 709-721, 2006.

[Pir04a] A. Pirzada, C. McDonald, “Kerberos Assisted Authentication in Mobile Ad-hoc Networks”, in *Proceedings of 27th Australasian Computer Science Conference (ACSC'04)*, pp. 41-46, 2004.

[Pir04b] A. A. Pirzada, C. McDonald, “Establishing trust in pure ad-hoc networks”, in *Proceedings of the 27th Australasian Computer Science Conference*, vol. 26(1), pp. 47–54, 2004.

[Poo00] R. Poor, “Gradient Routing in Ad Hoc Networks”, Media Laboratory Technical report 2000-01, Massachusetts Institute of Technology, 2000.

[Put03] R. Puttini, L. Me, and R. de Sousa, “Certification and Authentication Services for Securing MANET Routing Protocols”, in *Proceedings of the 5th IFIP TC6 International Conference on Mobile and Wireless Communications Networks (MWCN’03)*, October 2003.

[Rat00] K. Ratnam, S. Rangarajan, and A. T. Dahbura, “An Efficient Fault-Tolerant Location Management Protocol for Personal Communication Networks”, *IEEE Trans. on Vehicular Technology*, vol. 49(6), pp. 2359-2369, 2000.

[Rat01] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, “A Scalable Content-Addressable Network”, in *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM’01)*, pp. 161-172, 2001.

[Rob06] R. F. Erbacher, K. Christensen, and A. Sundberg, “Visual Forensic Techniques and Processes”, in *Proceedings of the 9th Annual NYS Cyber Security Conference Symposium on Information Assurance*, pp. 72-80, 2006.

[Row01] A. Rowstron, P. Druschel, “Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems”, in *Proceedings of the 2001 IFIP/ACM International Conference on Distributed Systems Platforms (Middleware’01)*, pp. 329-350, 2001.

[Sav00]: S. Savage, D. Wetherall, A. Karlin, and T. Anderson, “Practical network support for IP traceback”, in *Proceedings of the 2000 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM’00)*, pp. 295–306, 2000.

[Sek04]: V. Sekar, Y. Xie, D. Maltz, M. Reiter, and H. Zhang, “Toward a Framework For Internet Forensic Analysis”, in *Proceedings of the ACM SIGCOMM Hot Topics in Networks (HotNets’04)*, 2004.

[Sek06]: V. Sekar, Y. Xie, D. Maltz, M. Reiter, and H. Zhang, “A Multi-Resolution Approach For Worm Detection and Containment”, in *Proceedings of the IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'06)*, pp. 189-198, 2006.

[Sha04] M. A. Sharaf, J. Beaver, A. Labrinidis, and P. K. Chrysanthis, “Balancing energy efficiency and quality of aggregate data in sensor networks”, *The International Journal on Very Large Data Bases (VLDB)*, vol. 13(4), pp. 384–403, 2004

[Sha05] D. Sharma, V. I. Zadorozhny, and P. K. Chrysanthis “Timely Data Delivery in Sensor Networks using Whirlpool”, in *Proceedings of the 2nd international workshop on Data management for sensor networks (DMSN’05)*, Session: In-network processing and routing, pp. 53-60, 2005.

[Shi05] J. Shin, J. Kim, K. Park, and D. Park “Railroad: Virtual Infrastructure for Data Dissemination in Wireless Sensor Networks”, in *Proceedings of the 2nd ACM international workshop on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks (PEWASUN’05)*, Session: Technical papers, pp. 168-174 2005.

[Sno02]: A. C. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, Fabrice Tchakountio, Beverly Schwartz, Stephen T. Kent, W. Timothy Strayer, “Single-packet IP traceback”, *IEEE/ACM Trans. on Networking (TON)*, vol. 10(6), pp. 721-734, 2002.

[Sto01] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, “Chord: A scalable peer-to-peer lookup service for internet applications”, in *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM’01)*, pp. 149-160, 2001.

[Sto05] R. Stoleru, T. He, J. A. Stankovic, and D. Luebke, “A High-Accuracy, Low-Expense Localization System for Wireless Sensor Networks”, in *Proceedings of the 3rd international conference on Embedded networked sensor systems (SenSys’05)*, Session: Sensornet services, pp. 13-26, 2005.

[Tse07] C. H. Tseng, S. Wang, and K. Levitt, “DRETA: distributed routing evidence tracing and authentication intrusion detection model for MANET”, in *Proceedings of the 2nd ACM symposium on information, computer and communications security (ASIACCS '07)*, Session: Short papers, pp. 395-397, 2007.

[Vee97] M. Veeraraghavan, G. Dommety, “Mobile Location Management in ATM Networks”, *IEEE Journal on selected areas in Communications*, vol. 15(8), pp. 1437-1454, 1997.

[Ven00] L. Venkatraman, D. P. Agrawal, “A novel authentication scheme for ad hoc networks”, in *Proceedings of the 2000 IEEE Wireless Communications and Networking Conference (WCNC’00)*, vol. 3, pp 1268-1273, 2000.

[Wad05] A. Wadaa, S. Olariu, and L. Wilson, “Training a Wireless Sensor Network”, *Mobile Networks and Applications*, vol. 10(1), pp. 151-168, 2005.

[Wan01] W. Wang, I. F. Akyildiz, “A New Signaling Protocol for Intersystem Roaming in Next-Generation Wireless Systems”, *IEEE Journal on selected areas in Communications*, vol. 19(10), pp. 2040-2052, 2001.

[Xie05] Y. Xie, V. Sekar, D. Maltz, M. Reiter, and H. Zhang, “Worm Origin Identification Using Random Moonwalks”, in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 242-256, 2005.

[Ye02] F. Ye, H. Luo, J. Cheng, and L. Zhang, “A Two-Tier Data Dissemination Model for Large-scale Wireless Sensor Networks”, in *Proceedings of the 8th annual international conference on Mobile computing and networking (MOBICOM’02)*, Session: Sensor networks, pp. 148-159, 2002.

[Ye05] F. Ye, G. Zhong, S. Lu, and L. Zhang, “GRAdient Broadcast: A Robust Data Delivery Protocol for Large Scale Sensor Networks”, *ACM Wireless Networks (WINET)*, vol. 11(3), pp. 285-298, 2005.

[Zha03] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. Kubiatowicz, “Tapestry: A resilient global-scale overlay for service deployment”, *IEEE Journal on selected areas in Communications*, volume 22, issue 1, pp. 41-53, 2003.

[Zha07] Xiwei Zhao, V. Ganapathy, N. Pissinou, and K. Makki, “Revisiting Global Time Synchronization”, in *Proceedings of the 50th Annual IEEE GLOBAL COMMUNICATIONS CONFERENCE (GLOBECOM 2007)*, Nov. 26-30, Washington DC, USA, 2007.

[Zho99] L. Zhou, Z. J. Haas, “Securing Ad Hoc Networks”, *IEEE Network Magazine*, volume 13, issue 6, pp. 24-30, 1999.

[Zho01] S. Zhou, A. Seneviratne, and T. Percival, “A location management scheme for mobility support in wireless IP networks using session initiation protocol (SIP)”, in *Proceedings of 9th IEEE International Conference on Networks (ICON’01)*, pp. 486-491, 2001.

VITA

XIWEI ZHAO

Place of birth	Heilongjiang, P. R. China

Education

08/2004 - present	Doctoral Candidate, Electrical Engineering Florida International University Miami, Florida
08/1998 - 07/2001	Master of Science, Automation Tsinghua University Beijing, P. R. China
08/1987 – 07/1991	Bachelor of Science, Mechanics Zhejiang University Hangzhou, P. R. China

Recent Work Experience

Position: Research Assistant at Telecommunications & Information Technology Institute, Florida International University, Miami, FL, USA (Aug. 2004 - present)
Job Functions and Projects:

- Developing an extension of Distributed Hash Table to organize log files over mobile and anonymous network to support forensic analysis
- Developing secure solutions on node authentication over mobile network to fight back node-ID spoofing.
- Evaluation of global and local time synchronization schemes over mobile network
- Developing a self-adaptable clustering and routing scheme over Wireless Sensor Network
- Improvement of Location Management over GSM cellular network, by following anchor chain technology

Position: System Engineer at GUOAN GROUP, Beijing, China (Jul. 2001 - 2003)
Job Functions and Projects:

- System Architecture Design: Integrate sub-systems into a graphic platform
- Develop device drivers to collect data from field control units

Recent Publications

Journal and Book Chapters

1. Xiwei Zhao, K. Makki, N. Pissinou, "Self-adapt Routing for In-network Processing", International Journal of Smart Home, volume 1, issue 1, page 40-48, ISSN: 1975-4094, 2007.

2. Xiwei Zhao, K. Makki, N. Pissinou, "An Efficient and Robust Routing Protocol for Data Aggregation", Springer Berlin/Heidelbery, Series: Lecture Notes in Computer Science, volume 4138/2006, page 175-186, ISBN: 978-3-540-37189-2, 2006.

Technical Conference

1. Xiwei Zhao, V. Ganapathy, N. Pissinou, K. Makki. "Self-organized Forensic Support in MANETs". In Proceedings of 22th International Parallel and Distributed Processing Symposium (IEEE IPDPS 2008) Workshop, Apr. 14-18, Miami FL, USA, 2008.

2. Xiwei Zhao, V. Ganapathy, N. Pissinou, K. Makki, "Revisiting Global Time Synchronization", in Proceedings of the 50th Annual IEEE GLOBAL COMMUNICATIONS CONFERENCE (GLOBECOM 2007), Nov. 26-30, Washington DC, USA, 2007.

3. Xiwei Zhao, V. Ganapathy, N. Pissinou, K. Makki, "On the time synchronization of Multi-hop Ad hoc Networks", in Proceedings of the 2007 International Workshop on Theoretical and Algorithmic Aspects of Sensor and Ad-hoc Networks (WTASA'07), Jun. 28-29, Miami FL, USA, 2007.

4. Xiwei Zhao, K. Makki, N. Pissinou, "The Optimization of Location Management", in Proceedings of 4th IEEE Consumer Communications and Networking Conference (CCNC'07), Jan. 11th-13th, Las Vegas NA,USA, 2007.

5. Xiwei Zhao, K. Makki, N. Pissinou, "Sensing routing path for in-network aggregation", in Proceedings of 2006 International Workshop on Smart Home (IWSH'06), Nov. 10th-11th, Cheju Island (Jeju-do), Korea, 2006.