EFFICIENT NETWORK CAMOUFLAGING IN WIRELESS NETWORKS

A Dissertation

by

SHU JIANG

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

December 2005

Major Subject: Computer Science

EFFICIENT NETWORK CAMOUFLAGING IN WIRELESS NETWORKS

A Dissertation

by

SHU JIANG

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

Co-Chairs of Committee,	Nitin H. Vaidya
	Wei Zhao
Committee Members,	Riccardo Bettati
	Costas N. Georghiades
Head of Department,	Valerie E. Taylor

December 2005

Major Subject: Computer Science

ABSTRACT

Efficient Network Camouflaging in Wireless Networks. (December 2005)

Shu Jiang, B.E., University of Science and Technology, China;

M.E., Nanjing University, China

Co–Chairs of Advisory Committee: Dr. Nitin H. Vaidya
Dr. Wei Zhao

Camouflaging is about making something invisible or less visible. Network camouflaging is about hiding certain traffic information (e.g. traffic pattern, traffic flow identity, etc.) from internal and external eavesdroppers such that important information cannot be deduced from it for malicious use. It is one of the most challenging security requirements to meet in computer networks. Existing camouflaging techniques such as traffic padding, MIX-net, etc., incur significant performance degradation when protected networks are wireless networks, such as sensor networks and mobile ad hoc networks. The reason is that wireless networks are typically subject to resource constraints (e.g. bandwidth, power supply) and possess some unique characteristics (e.g. broadcast, node mobility) that traditional wired networks do not possess. This necessitates developing new techniques that take account of properties of wireless networks and are able to achieve a good balance between performance and security.

In this three-part dissertation we investigate techniques for providing network camouflaging services in wireless networks. In the first part, we address a specific problem in a hierarchical multi-task sensor network, i.e. hiding the links between observable traffic patterns and user interests. To solve the problem, a temporally constant traffic pattern, called cover traffic pattern, is needed. We describe two traffic padding schemes that implement the cover traffic pattern and provide algorithms for achieving the optimal energy efficiencies with each scheme. In the second part, we explore the design of a MIX-net based anonymity system in mobile ad hoc networks. The objective is to hide the source-destination relationship with respect to each connection. We survey existing MIX route determination algorithms that do not account for dynamic network topology changes, which may result in high packet loss rate and large packet latency. We then introduce adaptive algorithms to overcome this problem. In the third part, we explore the notion of providing anonymity support at MAC layer in wireless networks, which employs the broadcast property of wireless transmission. We design an IEEE 802.11-compliant MAC protocol that provides receiver anonymity for unicast frames and offers better reliability than pure broadcast protocol.

To my Grandfather, Li Shuwen.

ACKNOWLEDGMENTS

In my pursuit of education, I have received enormous support from faculty members, fellow students, friends and families.

Many thanks go to my co-advisor Dr. Nitin H. Vaidya for his continued guidance. He is full of innovative ideas and his enthusiasm has served as an inspiration. I am grateful for the tremendous time and energy he spent in steering my research, especially when I started exploring the new topics.

I am grateful to my co-advisor Dr. Wei Zhao for admitting me as a member of the NetCamo group and for providing financial support. He is an excellent role model through his dedication to success.

I would like to thank Dr. Riccardo Bettati and Dr. Costas N. Georghiades for serving on my committee.

I am fortunate to be among many good fellow students and friends. I would like to thank the following office mates and friends for many helpful discussions and for friendships: Dong Xuan, Yong Guan, Xinwen Fu, Wei Yu, and Shengquan Wang.

Finally, I thank my parents for their constant encouragement for my education. My wife, Nan, who shared all my experience of excitement and disappointment over the years, has always been the source of support. My life in this long journey is made more enjoyable by my little son, Nelson.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

CHAPTER I

INTRODUCTION

Along with the explosive growth of computer networks in the last decades, the security of information transmitted over the networks has become an ever-increasing concern among the network users. It is well known that there are various attacks that threaten the confidentiality, integrity and availability of the information. Researchers have developed innovative techniques, most notably cryptography, to detect, prevent or recover from security attacks and to protect the information. However, there is cost involved with applying any security mechanism, which tends to be proportional to the amount of protection provided. A higher level of security is usually achieved at a higher cost, such as, more consumption of resources, larger degradation of system performance, etc. When applying security to a network, we need a systematic approach. We should examine both security property and efficiency of a potential security design. The effectiveness and efficiency of a security mechanism is not only dependent on an adversary's ability to launch attacks, but can also be affected by the nature of networks where the mechanism is applied. It is very possible that a secure and efficient mechanism in fixed, wired networks becomes insecure or inefficient in mobile, wireless networks. In this dissertation work, we study a class of security techniques which provide network camouflaging services, demonstrate their respective limitations in wireless networks, and propose new techniques to overcome these limitations.

This dissertation follows the style of *Mobile Networks and Applications.*

A. General Network Security

In [83], a list of general security services that encompass most network users' requirements is given as below:

- Confidentiality: This protects the transmitted information from being accessed by unauthorized parties. The information here is usually the contents of messages exchanged between communicating parties. However, the existence of communication itself is also "information" that might attract the adversary's interest. This leads to the flow confidentiality service as an addition to the data confidentiality service.
- Authentication: This assures the receiver that a delivered message is indeed from the indicated sender and not from an impersonator.
- Integrity: This assures the receiver that a delivered message has not been altered by any party during transmission.
- Nonrepudiation: This ensures that neither the sender nor the receiver of a message can deny the transmission.
- Access Control: This ensures that only authenticated users can use the network.
- Availability: This ensures that a network maintains its normal operation under interruption or denial of service attack.

These services are provided by different security mechanisms. Cryptography is the most important technique that underlies most of the security mechanisms in use. For example, data confidentiality is commonly provided by encrypting the transmitted message with a key shared between the sender and the recipient of the message. How effective the service is depends on the security of the key. There are public key cryptography and symmetric cryptography, which have different requirements on key management and different performance/cost characteristics. Public key cryptography requires more computing resources and is slower than symmetric key cryptography, but does not have the extra cost involved in generating and distributing key, which is needed by the latter.

In this dissertation, we deal with a service that cannot be provided by cryptography alone. As we mentioned, flow confidentiality service protects information about communication itself, such as, the location and identity of the communicating parties, the frequency and length of messages being exchanged, etc. This information may yield clues as to the activity or intentions of unsuspecting network users, or may provide a covert channel for communication from an accomplice within the system [59, 17, 18]. Unfortunately, the so-called "flow" information cannot be encrypted. There are two cases. One class of information is traffic pattern information. It is derived from observations on network traffic, and does not exist, nor transmitted, in any concrete form. So we have nothing to encrypt. Another class of information is source and destination information existing in transmitted packet headers, which discloses the identity and location of network users that are communicating with each other. In this case, we cannot simply encrypt the packet headers, because most network protocols require them to be exposed in order to deliver the packets. To protect the flow information, we have to use unconventional techniques called *network camouflaging*.

B. Network Camouflaging

Network camouflaging is about hiding flow information, in particular, network traffic pattern and source/destination of transmitted data. Hiding the latter information is also called *anonymity.* The adversary obtains flow information through passive attacks, i.e., eavesdropping. Since it is difficult to prevent eavesdropping, the only effective way of hiding information is to present wrong information to the eavesdropper. The manipulated information, which is purposedly leaked to the eavesdropper, serves as a "cover" for the real information. Sending dummy messages is one of the most well-known methods for this purpose [93]. Other methods include rerouting, delaying, and reordering messages [59, 92]. In the following, we first discuss the adversary model and then introduce essential network camouflaging techniques.

1. Classification of Attacks

Absolute security is not practical. Security that can be achieved is closely related to the adversary's ability. So adversary model is a critical factor in any security design. It is important to understand what security attacks an adversary can possibly conduct to compromise security. Two classifications of security attacks are often used in literature: passive attacks versus active attacks [83], and external attacks versus internal attacks [54].

A passive attack is in the nature of eavesdropping on, or monitoring of, transmissions. The attacker tries not to interfere with the transmissions and attempts to be as "invisible" as possible. An active attack involves some modification of transmitted messages or the creation of fake messages. For instance, a masquerade attack takes place when the attacker impersonates as a legitimate user, sends and receives messages in its name; a replay attack occurs when the attacker retransmits previously recorded messages (e.g. a valid user authentication sequence) to produce an unauthorize effect (e.g. logging into a server); a modification attack simply means that some portion of a legitimate message is altered, or the messages are purposedly delayed or reordered; a denial of service (DoS) attack prevents or inhibits the normal user or management of communication facilities.

Passive attacks and active attacks present the opposite characteristics. Passive attacks are very difficult to detect, because they do not involve any alteration of data, whereas active attacks can be detected by many means. Passive attacks can be prevented, to certain degree, whereas it is quite difficult to prevent active attacks, which would otherwise require physical protection of all communication facilities and all paths at all times.

If an attacker launches attacks from compromised network nodes, it is called an internal attacker; otherwise, it is an external attacker. Once a node is compromised, the attacker acquires its internal states and cryptographic secrets. So cryptographic method is useless against internal attacker. A passive internal attacker may pose long-term threat to security, because it cannot be detected without exhibiting malicious behavior. In wired networks, in order to prevent internal nodes from being compromised, we can use firewall, access control and intrusion detection system, etc., to “fend off” attackers. To the adversary, even eavesdropping is not easy to do, because it must gain physical access to the communication facilities (e.g. wires, switches). But wireless networking dramatically shifts the balance in favor of the adversary. In wireless networks, the adversary can easily launch passive attacks (e.g., eavesdropping) and active attacks (e.g., jamming the link) as long as it is within radio transmission range of targeted nodes. The risk of node intrusion in wireless networks is also higher than in wired networks, because wireless networks are often deployed in hostile environment (e.g., battlefield) and wireless nodes tend to be mobile.

2. Hiding Traffic Pattern

Traffic padding is a common technique used to hide the traffic pattern. Assuming that an eavesdropper records all traffic passing through a link in attempt to measure

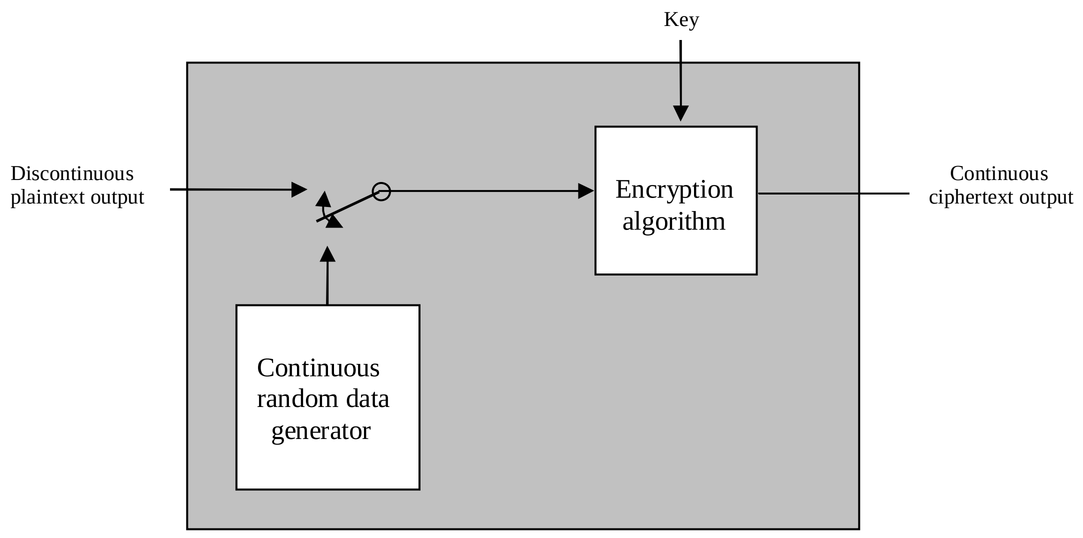
Key
Discontinuous plaintext output
Encryption algorithm
Continuous ciphertext output
Continuous random data generator

Fig. 1. Traffic padding encryption device [83]

the average traffic load on the link, the only way to hide the actual traffic load is to insert dummy traffic into the link such that the attacker observes an increased load. The prerequisite of this technique is that the attacker must not be able to distinguish dummy packets from real packets. This requires at least some portion of a packet to be encrypted (see Fig. 1).

Encryption of packet can be performed at two layers of the protocol stack. Link encryption is performed at the data link layer, by each intermediate node, before a packet is transmitted over a link. It hides the source and destination information, but allows loads on each link to be learned. End-to-end encryption is performed at the transport layer at the source node of a packet. It does not hide the source and destination information, and allows the attacker to learn traffic loads at the connection level. So different encryption types give different information to the attacker. With link encryption, each node can generate dummy packets to hide link traffic loads, as long as the nodes are secure. With end-to-end encryption, dummy packets can be generated to hide end-to-end traffic pattern.

3. Anonymity

Anonymous communication exists in many situations in the real life. For example, going to the library to make copies from books is anonymous. Buying a book in a bookstore and paying it with cash is anonymous as well. But their counterparts in the online world - browsing the Web for information and e-commerce - do not offer anonymity, since each IP packet discloses the IP addresses of both sender and receiver.

A lot of work has been done to solve the problem of anonymity in the network. Before we introduce each proposed technique, it is helpful to clarify the concept of *anonymity.* For this purpose, we refer to the nomenclature proposed by Pfitzmann and Kohntopp [65].

To enable anonymity of a subject, there has to be an appropriate set of subjects with the same attributes, called *anonymity set.* Anonymity is the state of being not identifiable within the anonymity set. Let the subject be a message. The sender of the message is anonymous if there is a set of possible senders. Similarly, receiver anonymity of a message is provided by a set of possible receivers. The larger the anonymity set is, the stronger the anonymity. So the anonymity set size is an important measurement for the anonymity that can be provided by a system. It is clear that the minimum size of the anonymity set is 2 (if there is only one user in the anonymity set, then there is no anonymity at all).

The following techniques differ in their approach towards establishing anonymity sets and therefore also differ in their possible field of applications.

a. Implicit Addresses and Broadcasting

A straightforward technique for receiver anonymity is broadcasting [67]. If a sender A wants to keep the recipient B of a message secret, he chooses additional pseudo recipients (e.g., C and D). Together with the real recipient B, these additional recipients form the anonymity set. The message is broadcasted to all members of the anonymity set. To identify the real recipient within the anonymity set, A uses an implicit address of B which can only be recognized by B and no one else. Unlike explicit address, an implicit address does not carry location information and cannot be used for routing purpose. [58] suggests to use the private keys as implicit addresses.

The technique has clear limit in a switching network due to the prohibitive broadcasting overhead. But in broadcast networks (e.g., satellite network, wireless network), it is a simple and low-cost solution for receiver anonymity [36, 51, 52, 25]. GSM networks already use this technique to hide the real identity of the users [10].

b. DC-network

The DC-network [15, 16] is a powerful technique for sender anonymity. The anonymity set is composed of all potential senders that exchange secret keys along a given key graph, i.e., a graph with the users as nodes and the secret keys as edges. To send a message, the sender A superposes (adds modulo 2) the message with the previously exchanged secret keys. Other users superpose in the same manner. All the messages of all users are transmitted to a receiver (or broadcasted if receiver anonymity is desired as well). The sum of these messages is just the message of A, because every secret key is added twice and cancelled. By this way, a message can be delivered without revealing the originator. According to [16], the technique provides an information-theoretic deterministic anonymity.

The application of the technique suffers from a high overhead. For every real message, $n - 1$ cover messages have to be transmitted. Therefore, [67] suggests to implement this scheme on a physical ring network.

c. MIX-network

The Mix-network is an anonymity technique that has been widely implemented in practical systems. Unlike broadcasting and DC-network, it shifts the burden of generating anonymity set from common users to a small set of special network nodes called *MIX nodes* or *MIXes*. This property makes it feasible in large scale networks since not every user has to participate in every communication. To send a message, the sender A encrypts the message and the recipient B's address with the public key of a MIX, and sends the message to the MIX. The MIX decrypts the message with its private key and forwards it to B. In addition to forwarding messages, the MIX takes actions to disallow an eavesdropper to link an output message of the MIX to any input message with certainty. Since the MIX receives messages from distinct users, all these users comprise the sender anonymity set of an output message. At the same time, the set of distinct users to which the MIX forwards messages becomes the receiver anonymity set of an input message. In another words, the eavesdropper knows either sender or receiver of a message, but not both. This is called *sender-recipient relationship anonymity* or *unlinkability* to differentiate from sender anonymity and receiver anonymity.

To fulfill the task of hiding the correspondence between input and output messages, a MIX must change the appearance (i.e., the bit pattern) of each message, break the timing link and the size link. The change of appearance is achieved by the cryptographic operations performed on the message. To break timing link, a MIX can use *batching* and *reordering*. Batching means that a MIX does not immediately forward every message. Instead, it collects messages and holds them in a buffer for a time. The flushing of the buffer can be triggered by a preset timer or by an event such as the arrival of suitably large number of messages. Reordering means that the messages leave a MIX in an order independent from their arrival order [50, 20]. One approach is to output messages in lexicographically sorted order. Another is to use random order. To break size link, all transmitted messages should have identical size. This can be achieved through *fragmentation* and *padding*.

So far we have considered a simple model in which every message is forwarded by only one MIX. If the MIX colludes with the adversary, then all information is exposed. To cope with this problem, the sender can choose to forward a message through multiple MIXes. To proceed, the sender first determines a MIX route, i.e., sequence of MIXes in forwarding the message, and then encrypts the message in a layered manner. Suppose that $M_1 \rightarrow M_2 \rightarrow \ldots \rightarrow M_n$ is the MIX route. Let $K(x)$ denote the encryption of a message x using a key K. The layered encryption of a message m to the recipient B can be denoted as

$$K_1(R_1, M_2, K_2(R_2, M_3 \ldots, K_{n-1}(R_{n-1}, M_n, K_n(R_n, B, m))))$$

where $K_1, K_2, \ldots, K_n$ are respective public keys of the MIXes and $R_1, R_2, \ldots, R_n$ are random strings. This message will be forwarded by the MIXes along the route. Each MIX strips off one layer of encryption from the message, throws away the random string, and sends the remainder of the message to the next MIX. The last MIX on the route delivers the original message m to the recipient B. With layered encryption, each MIX knows only its neighbor on the route. Revealing both sender and receiver of the message requires cooperation of all MIXes. In another words, the secrecy of any one MIX ensures the reliability of the entire route.

Batching and reordering of messages within MIXes increases the end-to-end delay

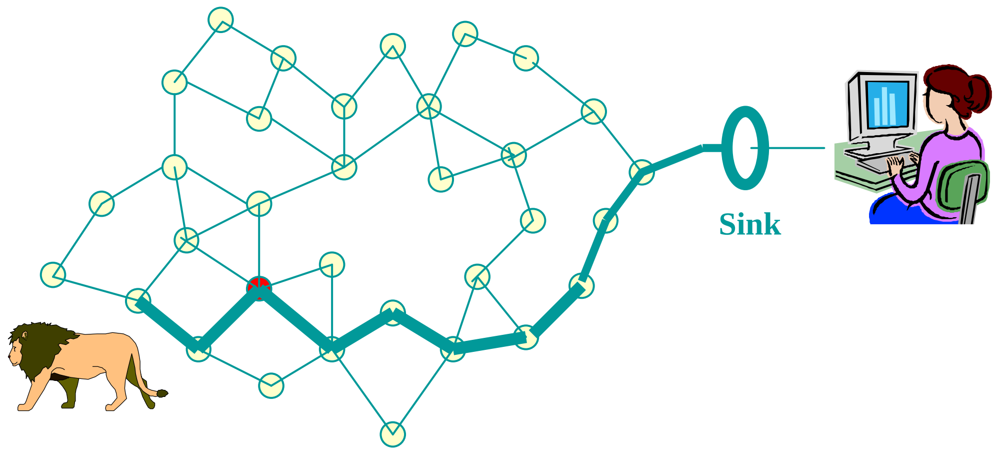

Fig. 2. An illustration of sensor network

for delivering the messages. So the technique was originally proposed for asynchronous message delivery system (e.g., email) [14, 33], which can tolerate substantial delay. How to apply the technique in connection-based system (e.g., web browsing), which has low-latency constraint, is still an unsolved problem. One approach is to send a large number of dummy messages to trigger frequent flushing of MIXes [66]. But dummy traffic reduces the utilization of network resources (e.g., bandwidth). Another option is to trade security in favor of performance. Hence, most of connection-based systems implemented so far [30, 73, 27, 78, 72] do not do any mixing as such.

C. Wireless Network Security

Wireless networking extends the reach of computer networks, leading to the prospect of “anywhere and anytime” communication. It is destined to have great impact on the future of humankind. There are many forms of wireless networks, both currently and in history. Recently, wireless sensor network and mobile ad hoc network have attracted the most interest from the research community.

A sensor network is composed of a large number of sensor nodes that are densely deployed in an area to monitor a phenomenon. In war, sensor nodes can be dropped from an aircraft into the enemy territory to monitor troop movements. In health, sensor nodes can be deployed to monitor patients and assist disabled patients. Some other applications include managing inventory, monitoring product quality, monitoring disaster areas, etc. Fig. 2 shows a sensor network deployed for habitat monitoring. Since the position of sensor nodes cannot always be engineered or predetermined, a sensor network has self-organizing characteristic, which means that sensor nodes actively find neighbors, establish communication links, and find routes to send the collected data back. The design of sensor network protocols and algorithms should be tailored to the unique characteristics of sensors, e.g., very low battery power, low computing capabilities, short communication distance, etc.

Another type of wireless network that possesses self-organizing capability is mobile ad hoc network (MANET) formed by a set of mobile users. Each user is equipped with a portable device such as laptop, PDA or Pocket PC, which is capable of sending and receiving data over a wireless medium, normally radio. Most of the current MANET designs are based on the assumption that all the devices share a single radio channel. Channel access is managed by a media access control (MAC) protocol (e.g., IEEE 802.11). Since each node's transmission range is limited due to signal attenuation, multi-hop routing of data may be necessary to enable communications between users, as shown in Fig. 3. The biggest challenge to the design of routing protocols in a MANET is the dynamic connectivity, caused by node movement, and variable network topology. Like sensor networks, MANET also has applications in many areas. In military, MANET technology can help soldiers in battle to exchange information in time and reduce casualties. In rescue operations, a MANET can be deployed in a short time and help the commander to coordinate activities. In a conference, attendees can form a MANET during meetings and upload/download files.

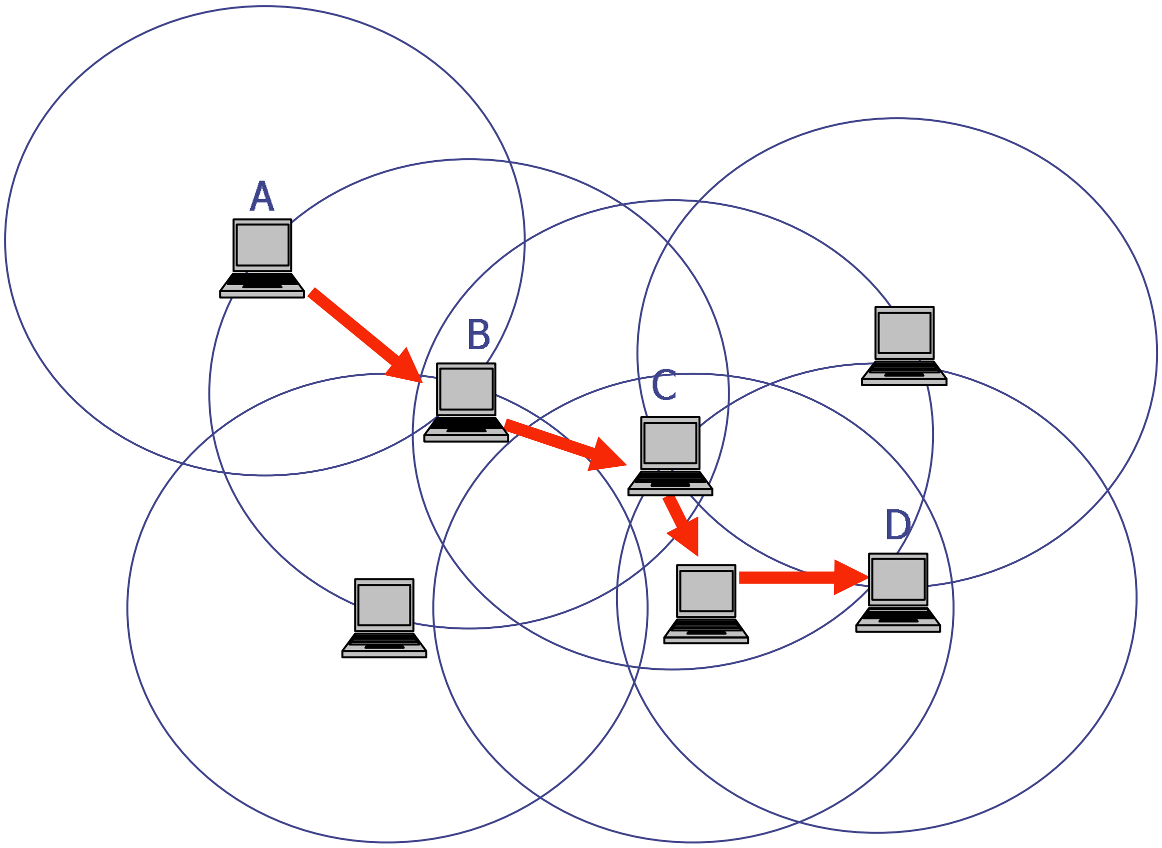

Fig. 3. An illustration of MANET

Wireless network security is an active research area at the present. Many schemes have been proposed to enhance security of both sensor networks and MANETs. These schemes cover a large spectrum of security issues, e.g., key management [101, 42, 99], secure routing [57, 39, 38, 60, 75, 3, 100, 23, 41, 61], authentication [82, 5, 40, 102], etc. Yet, the network camouflaging needs in wireless networks have not been addressed by previous work. In this dissertation, we address this particular area.

Sensor networks and mobile ad hoc networks have different camouflaging requirements. In a sensor network, the source and destination of data flows are often not a secret. The flows are from sensor nodes to a sink node (or gateway), which connects to users (directly or through a wired network). So hiding source/destination of each flow is usually not an issue here. But there might exist need of hiding traffic pattern. In chapter II, we show that distinct traffic patterns in a sensor network may be linked to different user interests (or tasks) and the information may be used for malicious purpose. A mobile ad hoc network is more dynamic than a sensor network. Nodes can move, join or leave a network at any time. Both network topology and membership are subject to change. Users tend to establish dynamic, short-lived connections between each other. There are needs of hiding the connection information, because it may be useful in deducing user activities. Consider a battlefield scenario where swarms of reconnaissance, surveillance and attack task forces form a tactical network to accomplish a mission. As the mission progresses, a large amount of communication occurs between the task forces and between members in each task force. If there is no anonymity support, the enemy may get important information from tracing the communications and from analyzing the communication pattern so that counterattacks can be prepared to thwart the mission.

In chapter II, we set the goal of hiding the temporal changes of traffic pattern in a sensor network, due to the changes of user interests. This requires to present a constant traffic pattern in the network. We examine potential approaches to reach the goal, namely, per source padding with end-to-end encryption and per link padding with link encryption. Since energy efficiency has the utmost importance in sensor networks, we attempt to give a quantitative comparison between the two approaches based on their abilities to extend system lifetime and to reduce system power consumption. Except that the minimum system power dissipation under source padding can be easily found by using the shortest path algorithm, the other three metrics, i.e. the respective maximum system lifetime under source padding and under link padding, the mininum system power dissipation under link padding, have to be found by formulating distinct flow optimization problems. Although we can use a LP (linear programming) solver to solve these problems, there exist more efficient approximation algorithms like ones proposed in this dissertation. The algorithms are based on a flow redirection technique. From numerical results, it is shown that the network can survive much longer or drains energy much slower with link padding.

In chapter III, we explore the design of MIX-networks serving mobile users. The mobile users comprise a MANET which provides physical connections between the MIXes. Like in wired networks, the sender of a message selects a MIX route to forward the message to its recipient. However, since the MANET does not have a fixed topology, it is a challenging task to provide stable data service through the MIX-network. Traditional design uses fixed or random MIX route based on the assumption that the network topology is fixed and known *a prior*, which is not appropriate in our settings. Hence, we propose a dynamic MIX route algorithm, which finds MIX routes based on the current network topology. In the algorithm, we address MIX advertising, anonymous MIX route discovery and MIX route update problem. The performance of the algorithm and its overhead are evaluated through simulations. It is shown that a dynamic MIX route has significantly better performance than a fixed or random MIX route, and the unit overhead decreases as the number of connections increases. The potential security weakness of using topology-dependent MIX route is also addressed.

In chapter IV, we propose a MAC-layer solution to the anonymity problem in wireless networks. Specifically, we design a MAC protocol which can hide the destination of a unicast transmission. It is based on two observations. First, a MIX-based anonymity system has inherent limitations on both security and performance, because the MIXes are obvious targets for active attacks and potential bottlenecks on the data paths. Second, wireless link transmission is in nature of broadcast, which can be utilized to provide receiver anonymity at a lower cost. However, a reliable transmission requires message exchanges between the sender and the receiver, such as acknowledgements. Under traffic analysis attack, these message exchanges could compromise anonymity. Our main contribution is to propose a batched, Selective Repeat (SR) ARQ scheme to improve transmission reliability, while maintaining receiver anonymity at the same time. Both security and performance of the proposed scheme are evaluated through analysis and simulations. It is shown that an anonymous broadcast scheme substantially increases the difficulty of traffic analysis attacks and provides fairly high untraceability in the presence of a small percentage of compromised nodes. Under various traffic loads, the proposed scheme presents significantly higher reliability than "pure" broadcast.

CHAPTER II

ENERGY EFFICIENCIES OF TRAFFIC PADDING SCHEMES IN A SENSOR NETWORK

In this chapter, we examine the energy efficiencies of traffic padding schemes in a sensor network. Our study is based on a specific application, i.e., hiding user interests in a multi-task sensor network. In such a network, as user changes interest, the traffic pattern in the network changes accordingly, which provides linking information to an eavesdropper. We suggest to hide the user's current interest by presenting a temporally constant traffic pattern. To achieve the goal, there are two traffic padding schemes available, namely, source padding and link padding. We demonstrate that a link padding scheme can achieve better performance than a source padding scheme with longer system lifetime and lower power dissipation. In the following, we first give the system model in section A, then define the metrics that we use to measure the energy efficiency of a traffic padding scheme in section B, which is followed by methods of evaluating each metric.

A. System Model

The sensor network in Fig. 2 is not optimal from energy efficiency perspective. In the network, all sensors route the collected data to a single sink node located on the periphery of the sensing area. Each sensor node does not only spend power on sensing and transmitting its own data, but also on forwarding other sensors' data. Thus, nodes drain energy at different rates, depending on their positions. As demonstrated in [35], sensors one-hop away from the sink deplete their energy quickly than other nodes. As a result, many sensor nodes will be unable to communicate with the sink and the network becomes inoperational. To overcome this problem, one of proposed

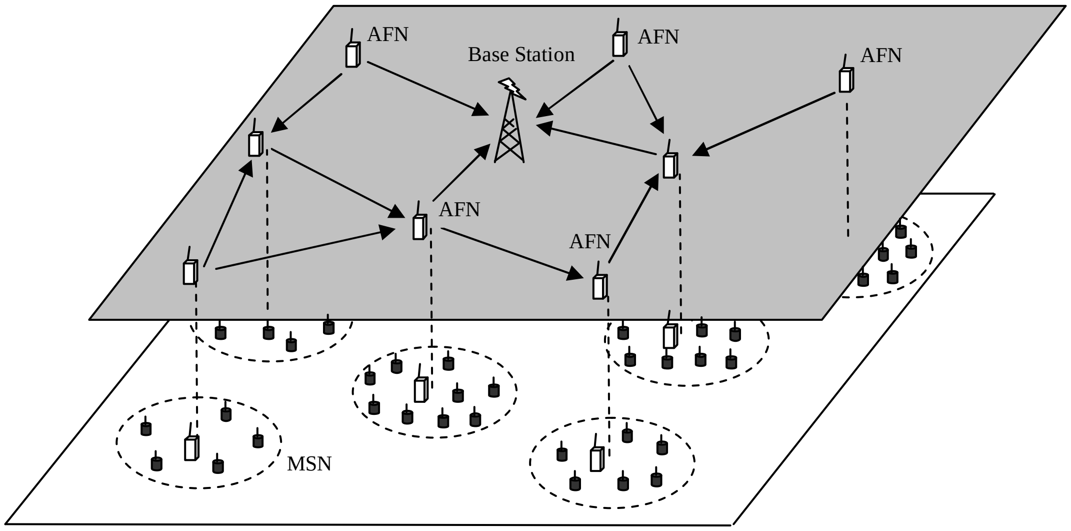

Fig. 4. Architecture of a hierarchical sensor network

solutions is a hierarchical clustered sensor network [37, 53]. Our study is based on this type of network.

Fig. 4 gives the architecture of a two-tier cluster-based sensor network. There are three types of nodes in the network, namely, *micro-sensor nodes* (MSNs), *aggregation and forwarding nodes* (AFNs), and a base-station (BS). The MSNs can be application specific (e.g., temperature sensors, pressure sensors, video sensors, etc.). They are deployed in groups (or clusters) at strategic locations for surveillance or monitoring applications. For each cluster of MSNs, there is one AFN, which serves as clusterhead. The MSNs are responsible for sending the collected data to the local AFNs. We assume that each MSN communicates directly with the AFN in its cluster. So MSNs do not have the responsibilities of relaying data. An AFN processes the data streams it receives from the MSNs within the cluster. Data aggregation (or "fusion") is necessary in sensor networks to reduce the amount of data transmitted to the base station. This is possible because a sensor network is data centric. For example, a user is interested in knowing the current temperature in a monitored area. After the user's interest is propagated to the sensors deployed in that area, the relevant AFN will receive a large number of temperature readings from the MSNs. Obviously, it is not wise to send all these readings to the user. What the AFN will probably do is to send the average value instead. The AFNs comprise an ad hoc network to relay each other's data. The base station is the sink node for data streams from all AFNs in the network. It serves as an interface between users and the sensor network. The AFN network is also utilized to propagate user interests from the base station to all AFNs.

The above architecture design relieves the MSNs of the burden of relaying data traffic, allowing longer time for performing their assigned duties. An AFN is expected to be provisioned with much more energy than a MSN. However, there is still need of conserving energy, because an AFN consumes energy at a substantially higher rate (due to wireless communications over large distances)[1]. Upon depletion of energy at an AFN, the coverage for the particular area under surveillance would be lost, although some of the MSNs within the cluster may still have remaining energy. To extend the lifetimes of the AFNs, we study energy efficient routing algorithm for the AFN network.

1. AFN Power Consumption Model

For an AFN, energy consumption due to wireless communication (i.e., transmitting and receiving) is a dominant factor in power consumption [1]. Therefore, in this investigation, we ignore the power consumption of data processing.

We assume that each AFN's transmitter has power control such that the transmission power can be adjusted based on the receiver's distance. Say AFN i transmits to AFN k at a rate of f_{ik} b/s. The power dissipation at the sender AFN i can be

[1]We assume that there is no energy constraint at the base station.

modeled as:

$$p_t(i,k) = c_{ik} \cdot f_{ik}, \tag{2.1}$$

where c_{ik} is the power consumption cost of radio link (i,k) and is given by

$$c_{ik} = \alpha + \beta \cdot d_{ik}^r \tag{2.2}$$

where α is a *distance-independent* constant term, β is a coefficient term associated with the *distance-dependent* term, d_{ik} is the distance between these two nodes, and r is the path loss index, with $2 \leq r \leq 4$ [70]. Example values for these parameters are $\alpha = 50nJ/b$ and $\beta = 0.0013pJ/b/r^4$ (for $r = 4$) [34].

The power dissipation at a receiver can be modeled as [70]:

$$p_r(i) = \rho \cdot \sum_{k \neq i} f_{ki}, \tag{2.3}$$

where $\sum_{k \neq i} f_{ki}$ (in b/s) is the rate of received data traffic at AFN i. A typical value for the parameter ρ is 50 nJ/b [34].

B. Problem Definition

In this section, we define the user security goal. Consider a sensor network with N AFNs. Within each cluster, there are sensors collecting data for different tasks. Each task corresponds with a unique user interest. For user interest j, we assume that the rate at which AFN i generates data is $g_i^j, i = 1, 2, \cdots, N$. When user changes interests, the traffic pattern in the AFN network changes as well. By observing traffic pattern currently in the AFN network, an adversary can learn what the user's current interest is. For example, a sensor network is deployed to monitor several important regions, but the security personnel can watch only one region at any time. By looking at the source of data traffic being transmitted in the AFN network, an adversary can know which region the user is currently watching and sneak into a region not being watched. Naturally, the following questions will be asked:

1. Is it possible to prevent the user's interest from being detected?
2. What is the cost of hiding user interests?

The answer to the first question is yes, and the solution is to present a constant traffic pattern in the AFN network, regardless of the user's current interest. In a naive approach, AFN i generates data at a rate $\max_j(g_i^j)$ all the time. If the current user interest is k and the corresponding data rate is lower than the maximal rate, then dummy data needs to be generated to increase the traffic load. Specifically, the dummy data rate is $\max_j(g_i^j) - g_i^k$. To fool the adversary, all data traffic must be encrypted such that dummy data is not distinguishable from real data. In this approach, each dummy packet is generated at an AFN and transmitted along the entire route to the base station as real data packets. So it is referred to as a *source padding* approach. The cost of this approach is typically high.

Now we consider another approach, called *link padding*, which has potential of reducing dummy traffic cost. It is based on the following observations:

1. User's current interest is broadcast to all AFNs in the network. If any AFN cooperates with the adversary, then any security scheme is useless.
2. Assuming that the adversary does not compromise any AFN, the traffic cover can be provided at the link level. This means that each AFN maintains a constant traffic load on all output links all the time. If the real traffic load on a link is lower than the expected level, then the sender node should generate

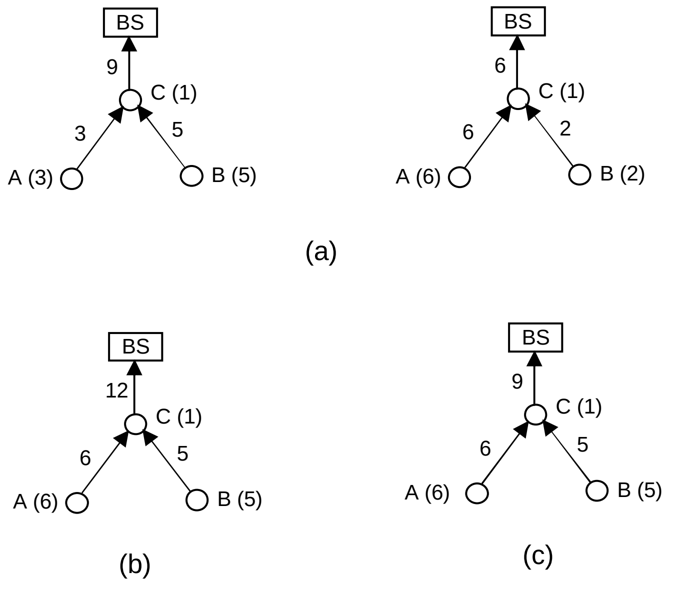

Fig. 5. (a) The traffic patterns corresponding with two distinct user interests; (b) The cover traffic pattern with source padding; (c) The cover traffic pattern with link padding.

dummy traffic. That is why it is called "link padding". Unlike source padding, each dummy packet traverses only one hop and is discarded by the receiver.

Link padding requires that all AFNs, instead of just AFNs who generate traffic, be responsible for maintaining the traffic cover. The benefit is that it can substantially reduce security costs, such as the amount of dummy traffic. In the following, we compare the two padding approaches by example. In Fig. 5(a), we give the traffic patterns corresponding with two distinct user interests. Nodes in the figure are AFNs. The numbers in parenthesis give the rate of traffic generated by each AFN in units per second. The number on each link gives the observed traffic rate on the link by an adversary in units per second. With source routing, each node maintains a constant traffic generation rate, which may be larger than needed by the current user interest. As shown in Fig. 5(b), node A generates data at a rate of 6 units per second. When the current user interest is the first one, 50% of traffic load generated is dummy data. Node C forwards all data from A to the BS without differentiation. With link padding, a constant traffic load is maintained on a per-link basis and dummy traffic is generated by each node at the link layer. As shown in Fig. 5(c), node C only forwards real traffic and discards dummy traffic from node A and B, and it generates new dummy traffic to maintain a constant load on the link to BS, which is lower than in source padding.

It is interesting to give a quantitative comparison between the two approaches. Traditional performance and cost metrics for traffic padding scheme are dummy traffic load, increased data latency, etc. But in sensor networks, energy efficiency is of utmost importance. So we use the following metrics in our study:

- System Lifetime T: Given the initial energy level of each AFN, the system lifetime is defined as the duration of time that elapses before any one AFN is depleted of energy, i.e., the minimum lifetime over all AFNs [12]. This definition is based on the notion that all AFNs are equally important in fulfilling the system function.

- System Power Dissipation R: Each AFN's power dissipation depends on the amount of traffic passing through the node. The system power dissipation is defined as the total power dissipation of all AFNs during the unit time interval. This metric measures the overall efficiency in energy use. A lower power dissipation level means that, using the same energy supply, more data traffic can be transmitted.

Simultaneous optimization of the two metrics may not always be possible. For instance, to reduce the power dissipation, data traffic should take minimum cost paths from the AFNs to the base station. Based on the AFN power consumption model, the closer two nodes are, the less energy is needed for transmitting and receiving data between them. This means that the close neighbors will be used in favor of distant neighbors as “next hops” in routing tables. As a result, some nodes may need to handle higher traffic loads and deplete energy supplies earlier than other nodes, which reduces the system lifetime.

Hence, the following optimality problems are distinct:

1. Given all traffic patterns of user interests, what is the maximum system lifetime of the cover traffic pattern, T_s, that a source padding scheme can achieve?

2. Given all traffic patterns of user interests, what is the minimum system power dissipation rate of the cover traffic pattern, R_s, that a source padding scheme can achieve?

3. Given all traffic patterns of user interests, what is the maximum system lifetime of the cover traffic pattern, T_l, that a link padding scheme can achieve?

4. Given all traffic patterns of user interests, what is the minimum system power dissipation rate of the cover traffic pattern, R_l, that a link padding scheme can achieve?

In the following sections, we discuss solutions for each problem. To facilitate evaluation, we make the assumptions below:

- The location of all AFNs are fixed.

- Traffic generated by an AFN can take different routes to the base station. The base station can run a centralized algorithm to determine the amount of traffic taking each route and notify each AFN. This is similar to traffic engineering in enterprise networks [2]. Based on global information, the base station can find the optimal traffic allocation with respect to an objective.

C. Optimization of Source Padding

In this section, we try to find T_s, the maximum system lifetime, and R_s, the minimum system power dissipation, with source routing. Consider a sensor network with N AFNs. The corresponding AFN network has $N+1$ nodes, including one Base Station (referred as BS). Assuming that every AFN has direct links to every other AFN and to the BS. The link from AFN i to AFN j is represented by (i, j), and the link from AFN i to the BS is represented by (i, BS). The power consumption required for transmitting unit data over a link is given by (2.2) and (2.3). Suppose that there are M user interests and the data generation rate at AFN i with respect to user interest j is $g_i^j (i = 1, 2, \cdots, N, j = 1, 2, \cdots, M)$. With source routing, each AFN i generates a mixture of real and dummy traffic at a constant rate g_i, which is the maximal possible rate of data traffic generated for any user interest, i.e.,

$$g_i = \max_j g_i^j \qquad (1 \le i \le N) \tag{2.4}$$

All data flows are routed towards the BS through multiple paths, subject to flow conservation constraints. Given the initial power supply at each node, system lifetime and system power dissipation are determined by traffic loads on all links. To maximize system lifetime, the optimal flow is the solution of the following LP (Linear Programming) problem:

$$\textbf{P1:} \quad Maximize \quad T \tag{2.5}$$

Subject to:

$$f_{iB} + \sum_{k \ne i} f_{ik} - \sum_{k \ne i} f_{ki} = g_i \qquad (1 \le i \le N) \tag{2.6}$$

$$\sum_{k\neq i}\rho f_{ki}T + \sum_{k\neq i} c_{ik} f_{ik} T + c_{iB} f_{iB} T \leq E_i \qquad (1 \leq i \leq N) \tag{2.7}$$

$$f_{ik}, f_{iB} \geq 0 \qquad (1 \leq i, k \leq N, k \neq i) \tag{2.8}$$

Variables:

f_{iB}	:	traffic rate on link (i, BS)
f_{ik}	:	traffic rate on link (i, k)

Constants:

g_i	:	traffic generation rate at AFN i
E_i	:	initial power at AFN i
c_{ik}	:	power consumption at AFN i for transmitting unit data to AFN k
c_{iB}	:	power consumption at AFN i for transmitting unit data to BS
ρ	:	power consumption at any node for receiving unit data

The set of constraints in (2.6) are the flow conservation constraints: they state that, the total data traffic transmitted by AFN i is equal to the total data traffic received by AFN i from other AFNs, plus the data traffic generated locally at AFN i. The set of constraints in (2.7) are the energy constraints: they state that, each AFN cannot spend more energy than the initial supply during the system lifetime. We can use a LP solver (e.g., CPLEX) to solve this problem, in polynomial time, and obtain the optimal solution. Or we can use the approximate algorithm in [12].

The system power dissipation is minimized when data traffic generated by each AFN i takes the minimal cost path to the base station with the cost of transmitting unit data over a link (j, k) being $c_{jk} + \rho$. We can use a number of shortest path algorithms (e.g., Bellman-Ford [19], Dijkstra [19]) to find the optimal paths.

D. Optimization of Link Padding

Suppose that there are M user interests in a sensor network with N AFNs. Each user interest j corresponds with a flow $\mathbf{f^j}$, $(1 \leq j \leq M)$, which satisfies the flow conservation constraints, i.e.

$$f_{iB}^j + \sum_{k \neq i} f_{ik}^j - \sum_{k \neq i} f_{ki}^j = g_i^j \qquad (1 \leq i \leq N, 1 \leq j \leq M) \tag{2.9}$$

$$f_{ik}^j, f_{iB}^j \geq 0 \qquad (1 \leq i, k \leq N, k \neq i, 1 \leq j \leq M) \tag{2.10}$$

where g_i^j is the data generation rate at AFN i, f_{ik}^j is the flow traffic rate on link (i,k), f_{iB}^j is the flow traffic rate on link (i,B). All are with respect to the j-th user interest.

With link padding, the apparent traffic load on a link is the maximal possible load of real traffic on that link with respect to all user interests. Let f_{ik} and f_{iB} represent the apparent traffic load on link (i,k) and (i,B), respectively. They must satisfy the following constraints:

$$f_{ik} = \max_j f_{ik}^j \qquad (1 \leq i, k \leq N, k \neq i) \tag{2.11}$$

$$f_{iB} = \max_j f_{iB}^j \qquad (1 \leq i \leq M) \tag{2.12}$$

The maximum system lifetime under link padding, T_l, and the minimum system power dissipation under link padding, R_l, are functions of f_{ik}'s and f_{iB}'s. Although we can use linear programming technique to find the values of the two metrics, it is an expensive solution because the number of independent variables in each LP problem, i.e. f_{ik}^j's and f_{iB}^j's, is $O(N^2M)$. In this work, we propose efficient approximate algorithms for each problem, based on the Flow Deviation (FD) method [26].

1. Maximization of System Lifetime

The Flow Deviation (FD) method is a general approach to flow optimization problem [26]. A FD-based algorithm is divided into two phases: the *initialization* phase and the *updating* phase. In the initialization phase, an initial solution of the problem is found. For example, each flow takes the minimal cost path. In the updating phase, a portion ϵ of a flow is shifted from its current path to a different path in a way that the objective function is improved. The former path is called a *giver* path and the latter is called a *taker* path. In a multi-commodity flow problem, both giver and taker are selected with respect to the same commodity. The flow redirection process is repeated until neither a giver path nor a taker path can be found. In this case, an approximate solution to the problem is found. It is possible that the algorithm returns a local optimal solution, instead of a global optimal solution.

Chang et al [12] uses the FD method to solve a similar system lifetime maximization problem. However, their problem is simpler than ours. Our problem is a special multi-commodity flow problem. It is different from the classical multi-commodity flow problem in that the link traffic flow is not *sum* of all commodity flows, but *max*. Modifying a commodity flow may not change the aggregate link flow or affect the system lifetime. We define *effective flow* to handle this problem.

When applying the FD method to solve our problem, the selection of giver path and taker path must always satisfy the following conditions:

1. Subtracting traffic from the giver path should increase the minimum lifetime of nodes on the path.

2. Adding traffic to the taker path might decrease the lifetimes of nodes on the path. Let T_{sys} be the system lifetime before the flow redirection action. None of nodes on the taker path should have shorter lifetime than T_{sys} after the action.

We now elaborate on how to find the giver, taker and ϵ.

We first form a subnetwork of the AFN network, $G^{(j)}$, which consists of all AFNs but only links with positive flows of a commodity j. Let $P_i^{(j)}$ be the set of all paths in $G^{(j)}$ from AFN i to the BS. For a path $p \in P_i^{(j)}$, we define its path length L_p as a vector whose elements are the lifetimes of all the AFNs in the path, and its critical node set S_p as the set of AFNs whose lifetimes are the minimal among all AFNs in the path. For example, if path $p \in P_i^{(j)}$ traverses nodes $i, n_1, n_2, \cdots, n_k$ before reaching the BS and $T_i, T_{n_1}, T_{n_2} \cdots T_{n_k}$ are respective node lifetimes, then $L_p = [T_i \quad T_{n_1} \quad T_{n_2} \cdots T_{n_k}]$ and S_p is the subset of AFNs whose lifetime is equal to $\min(T_i, T_{n_1}, T_{n_2}, \cdots, T_{n_k})$. Given two paths p and p', the length of path p is said to be shorter (longer) than the length of path p', if the smallest element of L_p is smaller (larger) than that of $L_{p'}$. In case they are the same, the next smallest elements are compared, and so on. If there are more than one smallest elements with the same value, then each one is compared separately. This is the so-called lexicographical comparison. Using a modified Bellman-Ford algorithm [19], the shortest length path from AFN i to the BS can be found. Note that the critical nodes of the shortest path have a lifetime equal to the current system lifetime. So the shortest path is a valid giver path, if subtracting some amount of traffic from the path can extend all critical nodes' lifetimes. In order to test whether the shortest path satisfies the condition [2], we define *effective load* of a commodity flow j on a link (x, y) as

$$\phi_{xy}^j = \begin{cases} f_{xy}^j - \max_{k \neq j} f_{xy}^k & \text{if} \quad f_{xy}^j > \max_{k \neq j} f_{xy}^k \\ 0 & \text{otherwise} \end{cases} \tag{2.13}$$

If $\phi_{xy}^j > 0$, then a reduction in f_{xy}^j would decrease the cover traffic load on the

[2]If there are more than one shortest path, each path is tested until a valid giver path is found.

link (x, y) and increase both node x's and node y's lifetimes. But if $\phi_{xy}^{j} = 0$, then reducing the link traffic load of this particular commodity flow has no impact on the link cover traffic load and does not change node x's or node y's lifetime. Thus, testing whether a path from AFN i to the BS can be a giver path for flow deviation is made simple. We just need to check whether the currently examined commodity flow has effective loads on links adjacent to the critical nodes such that all critical nodes' lifetimes can be increased by redirecting traffic away from the path. If a valid giver path does not exist, then we examine another commodity flow; Otherwise, we try to find a taker path and the amount of redirected traffic, ϵ, by taking the following steps:

1. Set ϵ to the minimum link traffic load of the commodity flow j among all links in the giver path.

2. Reduce f_{xy}^{j} by ϵ for all links (x, y) in the giver path.

3. Update f_{xy} based on equation 2.11 and 2.12 for all links (x, y) in the giver path, and update the lifetimes of all nodes in the giver path.

4. For each link (x, y) in $G^{(j)}$, test whether adding ϵ traffic load of the commodity flow j would reduce node x's lifetime below the current system lifetime T_{sys}. If yes, then it means that this link can not be in a valid taker path. Based on the testing results, we construct a subnet of $G^{(j)}$, denoted as $H^{(j)}$, which consists of all nodes and only links that fail the test.

   The details of the testing on a link (x, y) are given below:

   First, we define *effective* increase of cover traffic load on link (x, y), which is a function of ϵ, as:

$$\delta^j_{xy}(\epsilon) = \begin{cases} \epsilon & \text{if} \quad \phi^j_{xy} > 0 \\ max(f^j_{xy} + \epsilon - \max_{k \neq j} f^k_{xy}, 0) & \text{otherwise} \end{cases} \tag{2.14}$$

Second, we test whether the following inequality is satisfied:

$$\frac{1}{T_x} + \frac{(c_{xy} + \rho)\delta^j_{xy}(\epsilon)}{E_x} \geq \frac{1}{T_{sys}}, \tag{2.15}$$

where T_x is the current lifetime and E_x is the initial power supply of node x. In the above formula, $(c_{xy} + \rho)\delta^j_{xy}(\epsilon)$ gives an estimate of the increased energy consumption at node x due to the increased traffic load on the link (x, y). Note that the actual energy spent on receiving depends on which link the traffic comes from and is not known. So $\rho\delta^j_{xy}(\epsilon)$ is a heuristic estimate of the actual value. If it over-estimates the receiving energy cost, then the link (x, y) could be wrongly excluded from $H^{(j)}$. On the other hand, if it under-estimates the receiving energy cost, then the link (x, y) could be wrongly included into $H^{(j)}$. So, based on this test, the resultant subnet $H^{(j)}$ may not contain a valid taker path. This heuristic is employed in our implementation of the algorithm for numerical test. Alternatively, we can use another heuristic estimate, $\rho\epsilon$, which serves as an upper bound of the receiving energy cost. This heuristic has the property that, if AFN i and the BS are connected in $H^{(j)}$, then any path between them must be a valid taker path.

5. All paths from AFN i to the BS in the subnet $H^{(j)}$ are candidates for a valid taker path. We can use either Depth-First search algorithm or Breadth-First search algorithm to find such a path and test its validity. If a valid taker path from AFN i to the BS is not found in $H^{(j)}$, then the current ϵ value may be too high. In this case, set ϵ to its half and go to step 2.

6. The search process terminates when a valid taker path is found or when ϵ is less than a small threshold (0.1 bit/s in our simulations).

In the worst case, a Flow Deviation based algorithm may take exponential time to converge [8]. But it can be efficient in a probabilistic sense [13]. It is also possible that the proposed algorithm finds a local maximum, instead of the global maximum. In section E, we give numerical results on the performance of the algorithm.

2. Minimization of System Power Dissipation

In this section, we use the Flow Deviation method to solve another flow optimization problem related to link padding. It is to minimize the system power dissipation. By definition, the system power dissipation is the sum of node power dissipations, which turns out to be a linear function of traffic loads on all links, as below:

$$R_l = \sum_{(x,y)} (c_{xy} + \rho) f_{xy}, \tag{2.16}$$

where c_{xy}, ρ and f_{xy} are given by equations 2.2, 2.3, 2.11 and 2.12 respectively.

Using the Flow Deviation method, we must find an initial solution of the problem. This can be easily done by making all data traffic take the Minimal Cost Path from its source AFN to the BS, where the cost of each link (x, y) is defined as $c_{xy} + \rho$. Then we go to the updating phase in which all commodity flows are checked individually for possibility of flow deviation with the objective of reducing the system power dissipation. For a selected commodity flow, we need to find a giver path, a taker path and the amount of redirected traffic, ϵ.

Suppose that the commodity flow j is being examined. We first form a sub-network of the AFN network, $G^{(j)}$, which consists of all AFNs but only links with positive flows. To find a giver path, we set the weight of each link (x, y) in $G^{(j)}$ with positive effective load to be $c_{xy} + \rho$ and the weights of all other links to be zero. The definition of *effective load* is given in equation (2.13). Let $P_i^{(j)}$ be the set of all paths in $G^{(j)}$ from AFN i to the BS. For a path $p \in P_i^{(j)}$, we define its length as sum of weights of all links in the path. If the longest path from AFN i to the BS has positive length value, then it is a valid giver path, because removing traffic from the path must decrease the system power dissipation. The problem is whether we can find a taker path and the amount of redirected traffic, ϵ, such that the added traffic does not increase the system power dissipation above the current level. It proceeds as follows:

1. Set ϵ to the minimum link traffic load of the commodity flow j among all links in the giver path.

2. Reduce f_{xy}^j by ϵ for all links (x, y) in the giver path.

3. Update f_{xy} based on equation 2.11 and 2.12 for all links (x, y) in the giver path.

4. Set the weight of each link (x, y) in $G^{(j)}$ to be $(c_{xy} + \rho)\delta_{xy}^j(\epsilon)$, where $\delta_{xy}^j(\epsilon)$ is defined in equation 2.14.

5. Find the shortest weighted path from AFN i to the BS. If it is not a valid taker path, then set ϵ to its half and go to step 2.

6. The search process terminates when a valid taker path is found or when ϵ is less than a small threshold (0.1 bit/s in our simulations).

Like the algorithm in the last section, this algorithm may also take exponential time to converge. And, it is possible that a local optimal, instead of the global optimal solution, is found. In section E, we give numerical results on the performance of the algorithm.

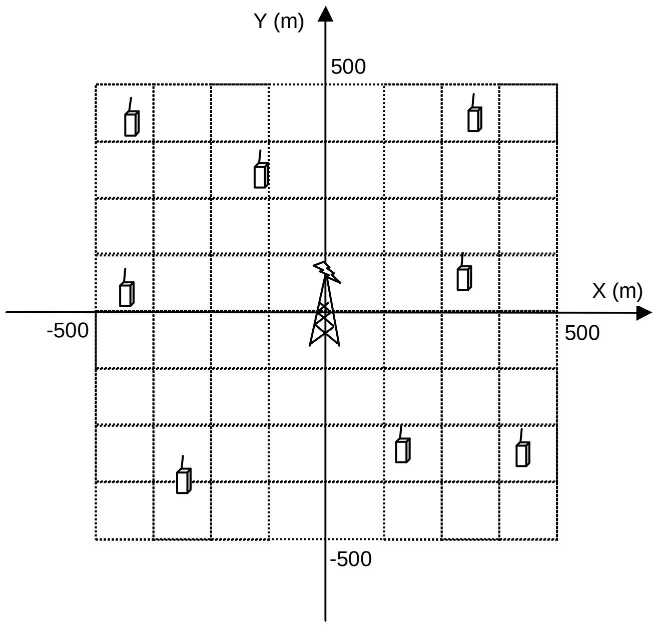

Fig. 6. A random AFN network

E. Numerical Results

In this section, we show numerical results obtained in simulations on random networks. A sample network is illustrated in Fig. 6, where a set of AFNs are randomly distributed over a 1000m x 1000m square area and the base station is located at the center of the area.

We generate networks with different number of AFNs. In sparse networks ($N = 10, 20, 30, 40$), we assume that nodes are fully-connected to ensure that each AFN has path to the base station. In other words, there is no upper limit on the transmit power at each node, and each node can send directly to the base station or any other node. In 50-AFN networks, we set each node's transmission range to 250m, which is sufficient to maintain full connectivity. The number of distinct user interests in a network is another parameter which varies between simulations. We assume that each AFN generates data for every user interest. The data generation rate is a random

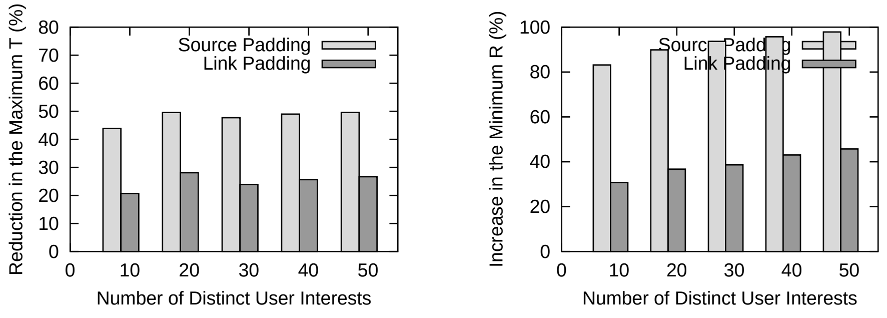

Fig. 7. Comparing the two approaches under different loads ($N = 50$)

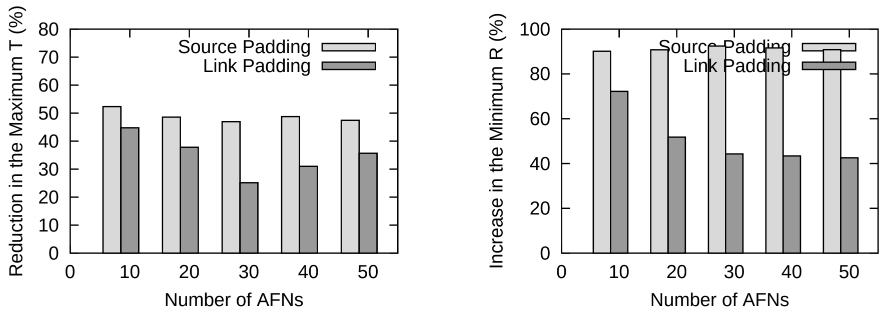

Fig. 8. Comparing the two approaches under different network sizes ($M = 20$)

number between 0 and 100 bit/s. The initial energy at each AFN is set to 50kJ. The power consumption behaviors for transmissions and receptions are defined in (2.2) and (2.3), respectively.

Compared to a network with no padding, either source padding or link padding reduces the maximal achievable system lifetime and increases the minimal achievable system dissipation inevitably. We compare the two approaches in degree of performance degradation. For this purpose, we need to find the maximal system lifetime and the minimal system dissipation for a given network and a set of traffic patterns, with no padding. We assume that each user interest has the same probability of being requested. The optimal value of each metric is found by solving two LP problems.

In Fig. 7, we compare the two approaches under different traffic loads in a 50-

AFN network. The number of distinct user interests varies from 10 to 50. Each value represents an average of 10 simulation runs. We can see that link padding reduces the maximal system lifetime by between 20%-30%, depending on the number of user interests, while source padding could reduce by 50%. It is also shown that source padding nearly doubles the minimum system power dissipation while link padding increases it by between 20%-40%. It is clear that source padding causes a greater degree of performance degradation than link padding.

In Fig. 8, we compare the two approaches under different network sizes. The number of distinct user interests is 20. Each value represents an average of 10 simulation runs. It is shown that the two approaches have close performances when the network size is small. This is due to a fewer number of distinct paths from each AFN to the BS in a small size network. In an extreme case, where each AFN has only a single path to send its data to the base station, the two approaches would produce the same results. When the distinct path set is large, data traffic from the same source will be distributed over a large number of paths. The more diverse the paths are, the lower the average traffic load on each link would be. This means that a link padding scheme can reduce the cover traffic load on each link.

F. Related Work

The importance of traffic flow confidentiality (TFC) to information security has been well acknowledged [45, 46]. It is becoming more important as government agencies and private companies are moving away from private networks and using public networks to meet their needs. TFC is frequently deemed as too detrimental to performance to be considered practical [93]. So a lot of studies are made on reducing the cost of this protection.

Newman-Wolfe and Venkatraman consider to hide traffic pattern in a network at the transport layer [59]. Their model assumes that end-to-end encryption is performed to protect the contents of transmitted data, but does not prevent an eavesdropper from learning traffic pattern information at the network address level, called *traffic matrix*. To prevent traffic analysis, they set a goal of presenting a neutral traffic matrix in the network, i.e., identical loads between all node pairs. In order to reduce the dummy traffic amount, they propose to reroute traffic from heavily loaded links to lightly loaded links. Rerouting causes non-local effects and cannot be done for one source-destination pair independently of the rest. Since all the rerouting decisions must be considered simultaneously, they formulate a linear programming problem to find the optimal solution. Guan et al. [32] extends this model in two aspects. One is that it allows the traffic to be rerouted through multiple intermediate nodes, instead of just one. Another is that it adds delay constraints required by real-time applications.

Adaptive traffic masking approach is proposed by Venkatraman [92], aiming at reducing the padding cost. Implemented at the transport layer, the rate of masked traffic increases and decreases as the rate of original traffic increases and decreases, thereby reducing the amount of dummy traffic. However, this approach may leak information about the original traffic and reduce the level of protection. Timmerman provides a remedy to the problem with a *Secure Dynamic Adaptive Traffic Masking* (S-DATM) scheme [89, 90]. In this scheme, statistical methods are used to detect and prevent (or limit) leaks of inference information which may occur when dynamic adjustments are allowed.

Radosavljevic and Hajek propose a technique of hiding traffic flow, which applies to Packet Radio Networks (PRN) [69]. In a PRN, a transmission schedule specifies at each time instant the set of nodes which are allowed to transmit. The purpose of a schedule is to prevent interference among transmissions from neighboring nodes. Normally, a schedule should be made based on the flow traffic demands of network nodes. In order to hide traffic flows, the authors propose to make a transmission schedule independent of traffic demands, using only the network topology as input. The schedule will be used indefinitely for all data transfers. If a node is scheduled to transmit, and has no data to send, it sends encrypted dummy messages instead. This scheme gives no information to the eavesdropper about the actual traffic demands, but can lead to a reduction in throughput, which is studied in the paper.

Fu et al. studied the implementation of traffic padding and have obtained counter-intuitive results [28]. In our study, we made a simple assumption about traffic patterns in the network by assuming that the interarrival time of packets in each traffic flow is constant. Based on this assumption, different traffic patterns are only distinguished by the average packet intervals. As we know, the reciprocal of packet interval is just the packet generation rate, or traffic load. The authors claim that constant interarrival times (CIT) of packets can only be implemented approximately in a real system. There must exist variance in observed packet interarrival times. Traffic padding with CIT is generally implemented by a timer-driven mechanism, which controls the transmission of dummy and real packets. For most of the modern operating systems, the real traffic flow will interfere with the timer's behavior. Heavier load incurs more interference, so the packet interarrival times of the overall traffic will have a correlation with the rate of the real traffic. If the adversary measures sample variance or entropy of the observed packet interarrival times, the probability of differentiating different traffic patterns could be greatly increased. In another words, CIT traffic padding is not as secure as it sounds.

CHAPTER III

A DYNAMIC MIX-NET IN MANET

In this chapter, we explore the design of MIX-net based anonymity system in a mobile ad hoc network. Our goal is to provide connection anonymity, i.e., hiding the source-destination relationship of end-to-end connections established between mobile nodes. As we know, MIX-net can serve this purpose. Assuming that we deploy a number of wireless MIX nodes in the area in which mobile nodes move around, the mobile nodes can establish encrypted connections with the MIXes and utilize the secure forwarding services that MIX nodes provide to achieve connection anonymity. Like mobile node, a MIX node has limited transmission range. So a mobile node may need to use the ad hoc routing mechanism to deliver packets to a MIX node. Similarly, data transfer between MIX nodes is also enabled by ad hoc networking. In a sense, the MIXes comprise an overlay network above the mobile ad hoc network. Fig. 9 illustrates the hierarchical architecture of such a network.

This design is ported from its counterparts (e.g., Onion Routing [30]) in wired networks. It has the advantages of easy deployment, easy management, independence from underlying network, etc., but suffers potential performance problem. Since all connections pass through one or multiple MIXes, congestion may easily build up around MIX nodes and cause long packet delay and high packet drop rate. In a MANET, new problems arise which exacerbate the situation. Due to node mobility, there is not a fixed network topology. If the network topology has changed and a connection still uses the same MIX route, it is possible that the packets traverse a long and detoured path to reach the destination node, which suffers a large delay and a high loss rate. Based on this observation, we propose adaptive MIX route algorithms which allow data packets of each connection to take suitable MIX routes

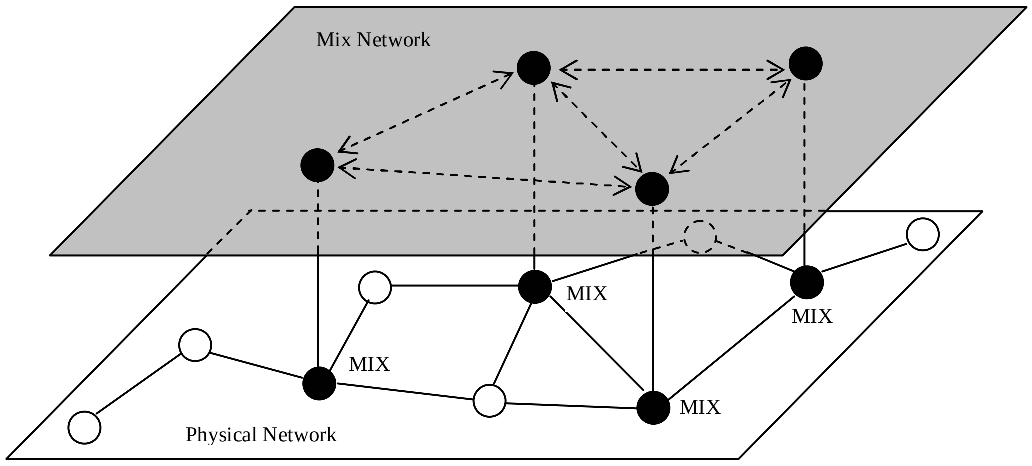

Fig. 9. The hierarchical architecture of a MIX-based anonymity system

based on the current network topology.

The rest of the chapter is organized as follows. In section A, we describe the system model. In section B, we introduce existing mix route algorithms. In section C, we present a simple MIX route algorithm, which finds the nearest MIX to any mobile node. Based on this information, the source node of a connection can always send data packets through the nearest MIX to it. In the algorithm, we provide two MIX discovery mechanisms: one based on on-demand MIX solicitations, another based on periodic MIX advertisements. In section D, we present a variable-length MIX route algorithm. This algorithm features destination-initiated MIX route advertisements. The performances of the algorithms are evaluated in section E via simulation, in comparison with algorithms for fixed MIX-nets. Our algorithms demonstrate significant performance improvements. In section F, we summarize related work.

A. System Model

Fig. 10 depicts the protocol stacks of both mobile nodes and MIX nodes used in this chapter. We elaborate the functionalities of each layer as below.

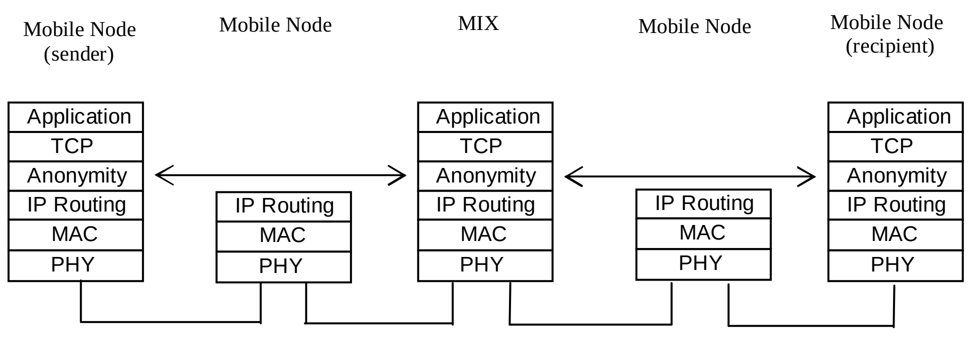

Fig. 10. The protocol stack at mobile nodes and MIX nodes

At the physical layer, all nodes communicate on the same shared wireless channel. For frequency hopping spread spectrum, this implies that all nodes have the same frequency hopping pattern, while for direct sequence spread spectrum, this implies that all nodes have the same pseudorandom sequence. We assume that all nodes have the same transmission range.

The MAC protocol coordinates access to the channel by different nodes so that conflicting transmissions are avoided. There are contention-based protocols and contentionless protocols. In MANET, the contention-based IEEE 802.11 MAC protocol [44] is the most pervasive one today.

The ad hoc routing protocol enables end-to-end communication between any network nodes. There are proactive routing protocols and on-demand routing protocols [74]. A proactive routing protocol attempts to maintain consistent, up-to-date routing paths from each node to every other node in the network. Any changes in network topology may cause network-wide propagation of routing information. So in a MANET, which characterizes frequent topology changes, proactive routing protocols may not be very efficient due to high routing overhead. A source-initiated on-demand routing protocol creates routes only when desired by the source node. When one node needs a route to another node, it initiates a route discovery process

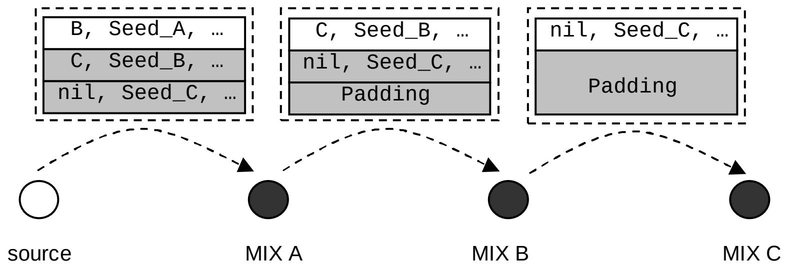

Fig. 11. The passing of onion along a MIX route

within the network. This process is completed once a route is found, or all possible route permutations have been examined. Once a route has been established, it is maintained by a route maintenance procedure until either the destination becomes inaccessible or until the route is no longer desired. Compared to proactive routing, on-demand routing has potentially much lower overhead, because there is no periodic routing updates. Proposed on-demand routing protocols include Dynamic Source Routing (DSR) [47], Ad Hoc On-Demand Distance Vector Routing (AODV) [63, 64], Temporally Ordered Routing (TORA) [62], etc.

The anonymity layer handles the anonymization of data packets received from the transport layer. Its functionality is split into two components: MIX route discovery and maintenance, and anonymous forwarding of data packets. The former will be elaborated in section B. We discuss the latter here. Suppose that the source node of a data packet already finds a MIX route that can be used to deliver the packet. It has two options regarding the packet encryption. If public key cryptography is adopted, it only needs to find the public keys of all MIXes on the route and then encrypt the packet using those keys. There are a number of key management schemes proposed for MANETs [101, 42, 99]. For simplicity, we assume that the public keys of all MIXes are published and known to all network nodes. If symmetric key cryptography is adopted, the source node needs to set up an anonymous connection, as in Onion

Routing, at first. Specifically, the source node passes a control packet, called *onion*, along the MIX route and propagates routing information and key generation material to all MIXes on the route. An onion is a multi-layered data structure. Each layer contains information (e.g., the next hop MIX address, key seed, expiration time) for one MIX, with the outermost layer targeting for the first MIX and the innermost layer targeting for the last MIX on the route. An onion is formed iteratively, innermost layer first. At each iteration, a new layer is added and is encrypted with the public key of the MIX that is intended to decrypt the layer. Fig. 11 illustrates the propagation of an onion along a MIX route. Each MIX that receives the onion obtains the key seed material, from which a decryption key is generated, strips one layer off and sends the remainder of the onion to the next MIX on the route. Since the source node knows all key seeds, it can generate the same keys as all MIXes, and use them for encrypting data packets. Regardless of which crypto-system is used, each data packet is encrypted with the keys of all MIXes on the route iteratively, similar to the construction of onion. The source node then tunnels the packet to the first MIX, which will decrypt the packet and forward it to the second MIX, and so on, until the last MIX tunnels the original packet to the destination node. The IPSec [49, 48] protocol provides a viable mechanism to facilitate this iterative tunneling process. Due to limitations on computational resources at mobile nodes, symmetric key cryptography (e.g., RC4 with 128-bit key) is usually preferred.

B. MIX Route Determination Problem

In a MIX-based anonymity system, the selection of MIX route for each packet has direct impact on packet latency. A MIX route is a logical route, not a physical route. The number of hops that a packet needs to take to reach its destination is the sum of the lengths of the shortest paths between consecutive nodes on the MIX route, starting from the source node and ending with the destination node. It is acceptable that an anonymized data packet takes a longer delay for delivery than an unanonymized data packet, which takes the shortest (or fastest) physical path. Some applications are more delay-tolerant than others. Generally, message-based systems, for asynchronous messages (e.g., email), can tolerate substantial delay, while connection-based systems, for real-time communication (e.g., web browsing, VoIP), have low-latency requirements.

In regard to MIX route determination, existing MIX-based anonymity systems fall into two classes. In one class, all MIX nodes in a system are administered by a single authority [85, 73]. In another class, all MIX nodes in a system are reachable from each other, i.e. fully connected [72, 27]. In both cases, the MIX-net topology is known, remains unchanged or changes very infrequently. Based on this assumption, the following MIX route algorithms have been proposed and often used:

- MIX cascade [14, 33]: This algorithm returns a fixed, predetermined MIX route for all data packets.
- Random MIX route [85, 24]: This algorithm returns a MIX route composed by randomly selected MIX nodes. A random route is normally constructed on a per-connection basis.

MIX cascade achieves the maximum security when at least one MIX is reliable, because all messages take the same route, and thus, all senders comprise the sender anonymity set and all receivers comprise the receiver anonymity set for all messages. However, the MIXes in a cascade are obvious targets for active attacks. If the adversary compromises all MIXes in the cascade, then the security of the entire system will collapse and all messages in the system will be exposed. From performance perspective, the mix cascade algorithm has no routing overhead involved with the finding of MIX route, but the data performance may suffer due to the fact that all connections share the same MIX route. In addition, the system is vulnerable to denial-of-service attack, because if any MIX in the cascade is forced out of duty, then the entire system halts or all nodes must reconfigure their MIX route settings.

Random MIX route alleviates the performance problem by distributing traffic loads among the MIX nodes. It is also resistant against active attacks in the sense that, if any MIX is compromised, only messages whose MIX routes pass through the MIX are affected. This reduces the damage inflicted upon the system by compromised MIXes.

However, both algorithms do not perform very well in dynamic MIX-nets. When network topology changes, the lengths of the shortest paths between network nodes are subject to change as well. If a fixed route is used, it is possible that the physical length of the route becomes very large or even the route becomes unavailable. If a random route is constructed without reference to the network topology, the route may not be the optimal. Our simulation results confirm this observation. In order to mitigate the performance degradation due to inferior MIX route, we try to develop new MIX route algorithms which are adaptive to changes in network topology.

C. Nearest MIX Algorithm

This algorithm is based on an intuitive idea: each data packet is forwarded by the nearest MIX from the source node. The distance between a mobile node and a MIX is defined as the length of the shortest path between them. It is possible that a mobile node has multiple nearest MIXes with identical distances. So each node maintains a Nearest MIX List (NML). When a data packet is generated, the source node can randomly select a MIX from its NML and forward the packet through the MIX. The algorithm is an extension of the Random Route algorithm mentioned above in that MIX routes are determined on a per-packet basis. Packets with the same source/destination may take different routes. When a node changes position, its distance to each MIX may change. Thus we need a dynamic mechanism for updating a node's NML in a timely manner. In the following, We present two mechanisms: one is based on MIX solicitation anycasting, another is based on MIX advertisement flooding.

1. MIX Solicitation Anycasting

In this mechod, a mobile node finds the nearest MIX node(s) by broadcasting a MIX Solicitation (MSOL) message, which is flooded over the network and triggers MIX Reply (MREP) messages from MIX nodes. A MSOL message has the following fields:

$$< MSOL, S, seqnum >$$

where S is the node who initiates the message and $seqnum$ is a unique sequence number generated by S. Initially, S broadcasts the message to its neighbors within the transmission range. If a mobile node receives the MSOL, it retransmits the message, subject to the constraint that the same message is retransmitted only once. A simple technique for duplicate detection is to record the sequence numbers of all previously transmitted MADVs from different sources. There is a retransmission jitter before the packet is put on the channel, which is a small random delay ($< 10ms$) and intended to resolve collisions. If any MIX node receives a MSOL, it does not retransmit the message, but unicasts a MREP message to the initiator of the search. In this sense, the MSOL is an anycast message. Each MIX only responds to the first MSOL it

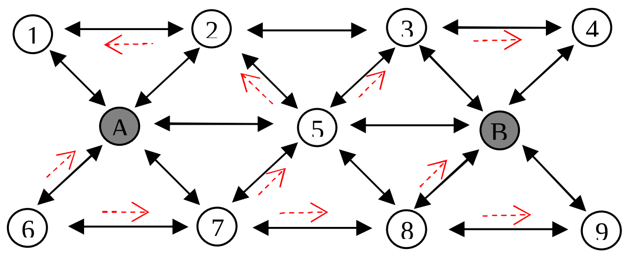

Fig. 12. An illustration of MSOL flooding

receives and ignores other instances of the same MSOL. A MREP message has the following fields:

$$< MREP, M \rightarrow S, seqnum >$$

where M represents the MIX node and *seqnum* is copied from the MSOL message. When S receives all MREP messages, it can update its NML based on distance information acquired from the network layer (e.g., the route cache maintained by DSR protocol or the route table maintained by AODV protocol). In other words, this algorithm requires inter-layer communication.

When a mobile node has no packet to send, it does not need to know the nearest MIXes. So a MIX discovery can be initiated on-demand. However, the source node of an on-going connection needs to initiate MIX discovery periodically during the connection time to maintain the "freshness" of NML. In each new MIX discovery, the MSOL message carries a new sequence number.

a. Example

Fig. 12 illustrates the flooding of a MSOL message in a network. In this example, node 6 initiates the MSOL message, which is received by node A and node 7 immediately. Node A is a MIX. So it does not retransmit the message, but replies a MREP message via unicast (not shown in the figure). Node 7 is a mobile node. So it retransmits the MSOL message, which is received by node 5 and node 8. Node A also receives node 7's transmission but this is a duplicate. So no action will be taken. In the figure, we only show arrivals of the first instances of the MSOL at each node. In this example, all nodes in the network receive the MSOL eventually. Node B, which is another MIX, also replies a MREP message to node 6. After receiving all MREPs, node 6 finds that A is the nearest MIX from it.

2. MIX Advertisement Flooding

MIX advertisement is an alternative mechanism for MIX discovery. In this mechanism, each MIX periodically broadcasts MIX advertisement (MADV) messages, which are flooded over the network and propagate distance information to mobile nodes. A MADV message has the following fields:

$$< MADV, M, seqnum, hop\_count >$$

where M is the MIX node who initiates the message, *seqnum* is a unique sequence number generated by M, which indicates the freshness of the message, and *hop_count* is the number of hops that the message has traversed before reaching a mobile node. Initially, M sets the *hop_count* to 0. Every time when a mobile node retransmits the message, it increases the *hop_count* value by one.

Each mobile node maintains a list of heard MIX nodes (referenced as HML) based on received MADV messages. For each MIX, the latest MADV sequence number and the minimum hop count to the MIX are recorded, based on which the Nearest MIX List (NML) can be easily derived. A mobile node only retransmits MADVs from the nearest MIX nodes, i.e., MIXes in its NML. The reasoning behind this heuristic is that MADVs from MIX nodes not in NML always have larger *hop_count* values than MADVs from MIX nodes in NML, before and after retransmissions. This rule restricts the flooding of each MADV to mobile nodes close to its source.

Before a mobile node retransmits a MADV, there is a small random delay, called retransmission jitter, which is intended to avoid collisions. A problem may arise due to this design. Consider a mobile node located several hops from its nearest MIX. The mobile node may receive multiple instances of a MADV from the MIX, with the same *seqnum* value but different *hop_count* values, because the messages have taken different routes to reach the mobile node. Due to retransmission jitter, it is very likely that one instance with a larger *hop_count* value arrives earlier than another instance with a smaller *hop_count* value. In this case, the mobile node would retransmit the MADV twice, because the later instance brings an uptated and "better" distance information. To reduce the number of retransmissions, one solution is to delay the retransmission of a received MADV when a mobile node can determine that other instances of the same MADV are likely to show up soon. Since each MIX broadcasts MADVs periodically, each mobile node can collect data about the length of time between arrival of the first and the arrival of the best instance of each MADV from each particular MIX, which is called *settling time*. Based on this data, a running, weighted average settling time of MADVs from a particular MIX is calculated, using the following formula, and helps to determine the length of delay before retransmitting a new MADV:

$$ST_i = \alpha \times ST_i + (1 - \alpha) \times t_i \tag{3.1}$$

In the above formula, ST_i is the average settling time of MADVs from MIX i, t_i is the measured settling time of the last sequence number, and α is a weighting parameter. In order to bias the calculation in favor of recent measurements, a smaller

α value must be selected. In our simulations, $\alpha = 0.875$. A mobile node delays for $2 \times ST_i$ before retransmitting a MADV from MIX i, if i is in the NML; the initial value of ST_i is 1 second. Note that this heuristic does not eliminate duplicate retransmissions of MADV completely. In rare cases, a new MADV may take a longer time than the retransmission delay to stabilize.

The time interval between two consecutive MADVs from each MIX is an important parameter in the system design. When node mobility is high, the NML of each mobile node may change very often. In this case, a small MADV interval is desired, which ensures that each mobile node updates its NML in time. When node mobility is low, the NML of each mobile node may remain stable for a quite long time. In this case, frequent MADV floodings are not necessary and harm the system performance due to increased overhead. In our simulations, we set the MADV interval to 5 seconds which is shown to be an appropriate value in moderately mobile networks.

While a mobile node is moving away from a MIX, it is possible that the node does not receive MADVs from the MIX any more. There are two possibilities. One possibility is that the mobile node is temporarily disconnected from the MIX. Another possibility is that none of the neighbors of the mobile node retransmits the MADVs from the MIX because there are other closer MIX nodes. In either case, the MIX entry should be removed from the HML and NML. This is implemented by setting expiration time for each entry. When a new MADV from a MIX is received, the entry's expiration time will be reset. In our simulations, the expiration time of a MIX entry in HML and NML is 15 seconds. As the lifetime of an entry is three times the MADV interval, a node is allowed to miss two consecutive MADVs from the MIX.

If a MIX node receives a MADV, it does not retransmit the message and simply discards the message.

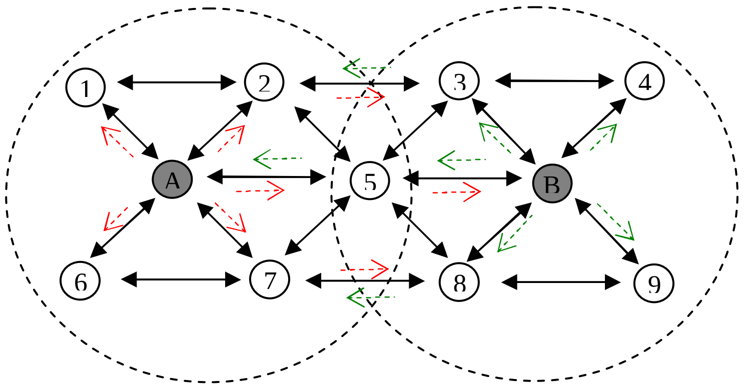

Fig. 13. An illustration of MADV flooding

a. Example

Fig. 13 illustrates the MADV flooding in a network. In this example, there are two MIX nodes. The retransmissions of MIX A's advertisements by mobile nodes are marked with red arrows and that of MIX B's are marked with green arrows. The two dashed circles mark the borders of flooded areas of each MIX's advertisements. Table I shows the contents of HMLs and NMLs of all mobile nodes. Node 5 has two MIXes in its NML, and it retransmits both MIXes' advertisements.

3. Comparison Between the Two Mechanisms

In this section, we compare the two MIX discovery mechanisms in terms of discovery latency and overhead. The MIX solicitation is an on-demand protocol. When a mobile node has data to send and does not know the nearest MIX(es) in its neighborhood, it initiates a new MIX discovery. Discovery latency is defined as the length of time from when the node transmits the MSOL until when all MREPs are received. It has two parts: the time spent on flooding the MSOL and the time needed for transmitting the MREPs. To reduce discovery latency, MSOL and MREP packets should have a higher priority in transmission than data packets and control packets generated

Table I. Contents of HMLs and NMLs

Node	HML	NML
1	{A}	{A}
2	{A,B}	{A}
3	{A,B}	{B}
4	{B}	{B}
5	{A,B}	{A,B}
6	{A}	{A}
7	{A,B}	{A}
8	{A,B}	{B}
9	{B}	{B}

by routing protocol (e.g., RREQ, RREP in DSR and AODV). Nevertheless, it has higher discovery latency than the MIX advertisement protocol, which ensures that each mobile node has up-to-date information about MIXes in its neighborhood all the time. The MIX solicitation overhead depends on the number of active source nodes. In a network consisting of n mobile nodes, it varies between 0 and n. Each mobile node retransmits a MSOL from a particular source once. So, in the worst case, the total number of MSOL transmissions that occur during a MIX discovery round is $O(n^2)$. The MIX advertisement overhead is independent of the user communication pattern, but is depends on the network topology. Each mobile node retransmits MADV's from the nearest MIXes. Suppose there are m MIX nodes in a network. The number of MIXes in a mobile node's NML varies between 0 and m. So, in the worst case, each mobile node retransmits MADVs from all MIXes, and thus, the total number of

MADV transmissions that occur during an advertisement interval is $O(nm)$. Since the number of MIX nodes is typically less than the number of mobile nodes, the MIX advertisement protocol is more efficient than the MIX solicitation protocol when the number of active source nodes is large.

4. Security Analysis

The Nearest MIX algorithm returns MIX routes each with a single MIX. Its effectiveness of hiding connection source/destination information heavily depends on the security of MIX nodes. If a MIX is compromised or colludes with the adversary, then all packets forwarded by the MIX will be exposed. Therefore, the following discussions are based on the assumption that all MIX nodes in a network are secure. We analyze possible attacks that can be mounted against the algorithm.

- Topology attack: This is a passive attack utilizing the network topology information. As we know, a distinctive characteristic of mobile ad hoc network is dynamic network topology. Generally, it is difficult for any node to obtain the current network topology information. Assuming that there exists an omnipotent attacker who can observe all node movements, based on the full network topology information, the attacker can easily rebuild the HML's and NML's in each mobile node. According to the algorithm, the source node of a data packet always chooses MIXes in its NML as forwarder. For each MIX, the attacker can construct a set of all mobile nodes who share the MIX in their NML's, which turns out to be the source anonymity set for all data packets forwarded by the MIX. Since the source anonymity set contains fewer nodes, it greatly increases the attacker's chance of revealing a packet's source/destination combination. To counter this attack, a small number of MIX nodes should be used in a network.

- Impersonation attack: This is an active attack that can be conducted by malicious nodes. In this attack, a malicious node claims to be a MIX, replies MREP messages or initiates MADV messages, depending on which MIX discovery mechanism is being used. Then unsuspicious source nodes would direct their traffic to this fake MIX. To counter this attack, an effective approach is to authenticate the creator of each control message and make sure that it is really from the claimed MIX. This can be implemented by using digital signature [83]. Digital signature is a public-key cryptographic mechanism. The creator of a message encrypts the entire message or a portion of the message with its private key and appends the ciphertext (i.e., signature) to the message. The receiver (or consumer) of a message decrypts the signature with the claimed creator's public key and verifies its correctness. If the creator is not compromised, then its digital signature is non-forgeable. Using this mechanism to authenticate the source of each control message in our algorithm, a node must have a prior knowledge of all trusted MIXes and their public keys. This information can be distributed by a centralized trusted server in the network [21].

- Tampering attack: This is another active attack. Unlike impersonation attack which targets at the creator of a control message, this attack targets at the control message directly. As we know, in the MIX advertisement scheme, each MIX broadcasts MADV messages periodically, which are retransmitted by mobile nodes. In normal operation, a mobile node increases the *hop_count* value in a MADV before retransmitting it, indicating that there is one more hop for the next receiver of the message to reach the originating MIX. However, a malicious mobile node may tamper the message with a higher sequence number or a smaller hop count. By doing this, the neighbors of the malicious node get wrong distance information and may add a distant MIX into their NML's. In the case that all MIXes are secure, this attack does not compromise the traffic anonymity and may only affect network performance by directing traffic through distant MIX nodes. But if there are compromised MIXes in the network, this attack has the effect of attracting data traffic to particular MIXes. To counter this attack, we can use the same message authentication scheme in the SEAD (Secure Efficient Distance Vector Routing) protocol [38]. This scheme uses efficient one-way hash chains [83] to authenticate *seqnum* and *hop_count* in a MADV. An one-way hash chain $h_0, h_1, \cdots, h_n$ is constructed by using an one-way hash function H repeatedly, starting with an random value h_0, such that $h_i = H(h_{i-1}), 0 < i < n$. It has the property that, if $i < j$, then h_j can be derived from h_i, but the contrary is not true. At the network setup time, each MIX computes an one-way hash chain and broadcasts the last element, i.e., h_n, to all nodes. Then, every time when a MIX generates a new MADV message, it appends a hash value to the message, which serves as the message authenticator. For a sequence number i, the hash value is h_{n-mi}, where $m-1$ is the maximum network diameter. When a mobile node needs to retransmit a MADV message, it increments the *hop_count* value by 1 and updates the message authenticator to a new hash value. For a sequence number i and a hop count j, the hash value is h_{n-mi+j}, which is next to the old hash value in the chain. Any receiver of the MADV message can verify its authenticity by applying the hash function H for $mi-j$ times on the embedded hash value and examining the final result, which should be h_n. Since an attacker can not compute a hash value with a smaller subscript than the current one, he is not able to modify a MADV to have a higher sequence number or a smaller hop count value, which would require a hash value in the chain with a smaller subscript to authenticate.

D. Destination-Initiated MIX Route (DIMR) Algorithm

In this section, we present the design of a new MIX route algorithm. It is different from the Nearest MIX algorithm in that the destination information is factored into the route selection process. In the algorithm, the destination node of a connection initiates a route advertisement process, proactively, to propagate route information to the source node. So it is called a Destination-initiated MIX Route (DIMR) algorithm. Compared to the Nearest MIX algorithm, the DIMR algorithm has improved security because it can be configured to find MIX routes with multiple MIXes.

We now elaborate on the MIX route discovery and maintenance process in the DIMR algorithm. It is divided into three phases:

1. *Anonymous route request phase*: When a mobile node (source) needs to find a MIX route to send its data packets to another mobile node (destination), it first sends an anonymous Route Request (RREQ) message to the destination node via a random MIX route. The RREQ message is encrypted and packaged in a data packet. In other words, there is no discernable identification in the packet which tells an eavesdropper that this packet carries a RREQ message. Otherwise, the eavesdropper would be aware of a node's desire of initiating communication with some other node. The random MIX route can be formed by MIXes selected from the node's Nearest MIX List(NML) or Heard MIX List(HML). To acquire NML and HML information, we assume that all MIXes broadcast MADV messages, as in the Nearest MIX algorithm.

2. *Registration phase*: When the destination node receives the RREQ, it selects a set of MIXes from its NML or HML and sends node registration requests (NREG) to each of them individually. Each NREG carries a sequence number, which indicates the freshness of the message. The NREGs sent to different

MIXes at the same time would bear the same sequence number. The destination node registers at multiple MIXes to provide the source node with more choices of MIX routes.

3. *Route advertisement phase*: Each MIX maintains a registration table, which lists all mobile nodes whose registration requests were recently received. Each registration table entry contains the following information about a particular node:

   - The node's address;
   - The number of hops required to reach the node;
   - The latest NREG sequence number;

   The DIMR algorithm requires each MIX to broadcast all the entries in its registration table to its current neighbors, periodically. The entries are then flooded throughout the network, along a shortest path tree rooted at the source MIX. Assuming that the network is connected, every node will receive the entries eventually. The path that the information travels is a mixture of MIX nodes and mobile nodes. There is at least one MIX on the path, i.e., the source of the entries. From the path, a mobile node can acquire a MIX route to another mobile node who is currently registered at the MIX. This is why the process is called *route advertisement*.

   During a route advertisement, a MIX's registration table entries are carried by a number of Route Advertisement (RADV) packets. Each RADV has the following fields in it:

$$< RADV, M, seqnum, mix\_route, regtab >$$

where M is the MIX node who initiates the advertisement, $seqnum$ is a unique sequence number generated by M, mix_route is a sequence of MIX nodes that the packet has traversed, and $regtab$ is a copy of M's registration table. Initially, the mix_route field contains one MIX, i.e., M. When a mobile node retransmits the packet, it does not change any field; but when a MIX node retransmits the packet, it appends itself to the end of mix_route. Each node retransmits a RADV only once. This is ensured by checking a list of previously retransmitted MADV packets, which is maintained at each node. Although a node does not retransmit duplicate RADV packets, it may learn distinct MIX routes from them to reach nodes listed in $regtab$. In a node's route cache, each route is tagged with the destination node's sequence number learned from the registration entries. Routes with more recent sequence numbers are always preferred as needed for forwarding data packets. Of the routes with the same sequence number, those with smaller lengths will be used more often.

In order to reduce the amount of propagated information, we can define two types of route advertisements. One will carry all entries in the source MIX's registration table, called a "full dump". The other type will carry only information changed since the last full dump, called an "incremental" advertisement. By design, an incremental updates should fit in one RADV packet, subject to the size limit set by network protocol. Full dumps can be transmitted relatively infrequently when no movement of registered nodes is occurring. When movement becomes frequent, and the size of an incremental approaches the size limit of RADV packet, then a full dump can be scheduled (so that the next incremental will be smaller).

As network topology may change over time, it is necessary to update the MIX

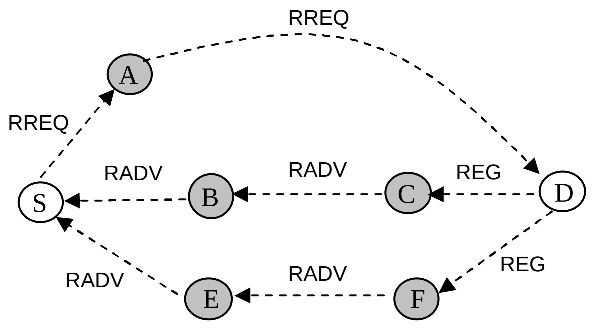

Fig. 14. Route discovery in the DIMR algorithm

route cache at each mobile node based on the current topology. The DIMR algorithm requires that, when a destination node detects changes in its NML or HML, it sends registration requests to a new set of selected MIXes. The NREG packets carry a new sequence number. Any MIX who receives a new NREG should update its registration table properly, and schedule a route advertisement. A node does not need to send specific "deregistration" information to the MIX(es) with which it was previously registered, because when the old registrars will receive RADVs flooded by the MIXes where the node is currently registered and learn the change of registration from the new sequence number associated with the node in the *regtab* field.

a. Example

Fig. 14 illustrates the route discovery process we just described. It is shown that node S sends a RREQ message to node D through MIX A, node D registers at MIX C and F, and the two MIXes initiate independent route advertisements. As a result, node S receives two MIX routes to reach the destination node D.

1. Properties of the Algorithm

The routing mechanism in the DIMR algorithm is a proactive process. From the route advertisements, each mobile node acquires MIX routes to all destination nodes of active connections. This means that the minimal route acquisition latency is zero, when a node's new connection shares the same destination with another currently active connection.

In a sense, the DIMR algorithm is an extension of the Nearest MIX algorithm. It uses the Nearest MIX algorithm to deliver RREQ messages and NREG messages. Both RREQ and NREG are unicast messages and incur small overhead. The main overhead is in route advertisements. In a network which consists of N mobile nodes and M MIXes, during a route advertisement interval, the MIXes generate M RADV messages and each RADV is flooded to all network nodes. Therefore, the total number of RADV transmissions is $O((N + M)M)$.

2. Security Analysis

Unlike the Nearest MIX algorithm, the DIMR algorithm is resistant to compromised MIX nodes, because it returns MIX routes with multiple MIXes. The connection anonymity is protected unless all MIXes on a route are compromised. In the above description of the algorithm, the MIX route length, i.e., the number of MIXes in a route, is variable and the minimum length is one. With minor modifications, we can make the algorithm always return MIX routes with a minimal length, say $\lambda (> 1)$. It works as follows. When a mobile node or a MIX node receives a RADV, it retransmits the message as long as the MIX route contains less than λ MIX nodes; otherwise, it retransmits the message for only once. In another words, the uniqueness of a RADV is determined by combination of sequence number and MIX route length, instead of just sequence number. Of course, the improved security is not achieved without cost. The larger λ is, the longer the path that each data packet needs to take.

In the following, we analyze possible attacks that can be mounted against the algorithm. We only consider attacks against connection anonymity.

- Topology attack: This is a passive attack utilizing the network topology information. In the DIMR algorithm, the source node of a connection receives MIX route updates from RADV messages flooded by the MIX nodes periodically. For a long-standing connection, it is possible that different MIX routes are used to deliver data packets. This is caused by frequent topology changes during the connection lifetime. If the attacker has full network topology information, an intersection attack can be conducted to reveal the destination node of a long-standing connection. It works as follows. Based on the topology information, the attacker can find all MIX routes that the source node will use with respect to each destination node. Then the attacker can observe which MIX the source node sends packets to. This first MIX information can be used to narrow down the set of possible destinations of the packets. Soon later when the network topology changes and the source node uses a different MIX route for the same connection, the attacker can derive a new set of possible destinations. The intersection of the two sets must contain the destination of the connection. If the attacker is lucky, the intersection set may contain only one node, which reveals the connection destination immediately. To prevent this attack, a perfect solution is that the source node sends real or dummy packets to all MIX nodes all the time so that the attacker gets no useful information. But it is also a very costly solution. A less costly solution is that the source nodes uses multiple MIX routes to send the data traffic and each route starts with different MIX nodes. This will complicate the attacker's task and reduce the possibility of being compromised.

- Tampering attack: This is an active attack launched by malicious nodes through modifying the contents of RADV messages. The objective of the attacker is to attract the source nodes of data traffic into using a compromised MIX route. A successful tampering attack must satisfy two conditions: (1) The advertised MIX route (in *mix_route* field) is a fully compromised MIX route; (2) The destination of an anonymous connection is listed in the *regtab* field. The first condition must be met because only a fully compromised MIX route, which consists of compromised MIXes, can reveal the connection source/destination combination. The second condition must be met because a connection source uses an advertised MIX route only when the connection destination is listed. So, to defend this attack, we can take the following two approaches. One is to set a large "minimal route length", i.e., λ in above discussions, and make it difficult to construct a fully compromised MIX route. Another is to add an authenticator to each entry in the *regtab* to prevent that the attacker fabricates node registration requests. In specific, each mobile node who sends RREG message must include a digital signature in it and each MIX maintains signed registration requests in its registration table. When a MIX node generates RADV messages, it copies the signed node registration requests into *regtab* fields. When a mobile node or a MIX receives a RADV message, it can verify the authenticity of each entry in the *regtab*. To protect the *seqnum* field from being tampered, the creator of each RADV message should also sign the message.

E. Performance Evaluation

1. Simulation Model

The simulations are implemented using the ns-2 network simulation package [7]. The simulation environment models a MANET of 50 mobile nodes and a variable number of MIX nodes. At the beginning of the simulation, all nodes are randomly placed in a 1000m by 1000m area. MIX nodes do not change positions during the simulation. Mobile nodes roam in the area, following the random way-point model of [47] with no pausing. When the simulation starts, each node randomly picks a destination and moves towards it with a random constant speed. After a node reaches its destination, it selects a new destination and starts to move towards it at a newly selected speed. The maximum node speed is a parameter for measuring node mobility. Each simulation is executed for 600 seconds.

The network stack of each node consists of an anonymity layer, a network layer, a link layer, an ARP module, an interface priority queue, a MAC layer and a network interface. The radio transmission range is approximately 250 meters. Each network interface transmits data at a rate of 2 Mbits/sec. IEEE 802.11 is used as the MAC protocol. The DSR protocol is used as the routing protocol.

We implemented the Nearest MIX algorithm and the DIMR algorithm above the ad hoc routing protocol. Table II lists the values of parameters in the simulations. For comparison purpose, we also implemented a fixed MIX algorithm and a random MIX algorithm. In the fixed MIX algorithm, all connections use the same MIX to forward data packets, while in the random MIX algorithm, each connection uses a MIX randomly selected at the connection setup time. Each MIX node adds a random delay between 0 and 50 milliseconds before transmitting a data packet. This simulates a stop-and-go MIX [50]. However, we ignore the processing delay for each packet

Table II. Parameter Values in Simulations

Parameter	Value
MIX solicitation interval	5 secs
MIX advertisement interval	5 secs
MIX advertisement lifetime	15 secs
Route advertisement interval	15 secs

introduced by cryptographic operations.

In all simulation runs, CBR traffic flows are injected into the network from source nodes. The size of data payload is 512 bytes. Data packets are generated at each source at a rate of 4 packets per second. The source nodes and destination nodes of all connections are randomly chosen from the mobile node set.

To evaluate the performance of a MIX route algorithm, we collect the following information:

- *Packet delivery fraction*: This is the ratio between the number of data packets delivered and those originated by the sources.
- *Average end-to-end data packet latency*: This includes the time for finding MIX route.
- *Control overhead*: This is the total number of control packets generated during the simulation run.

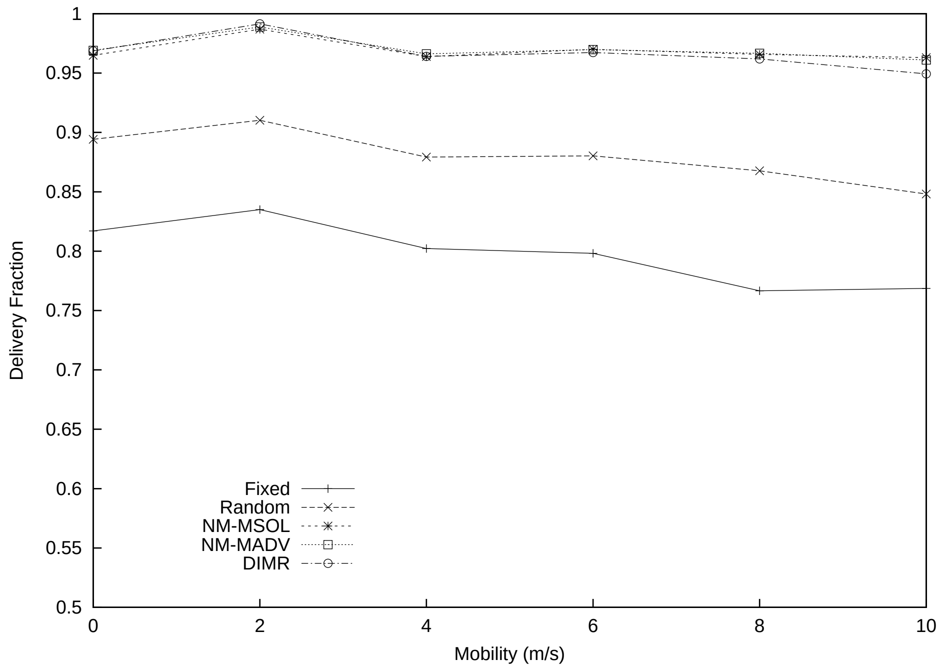

Fig. 15. Delivery fraction vs. mobility

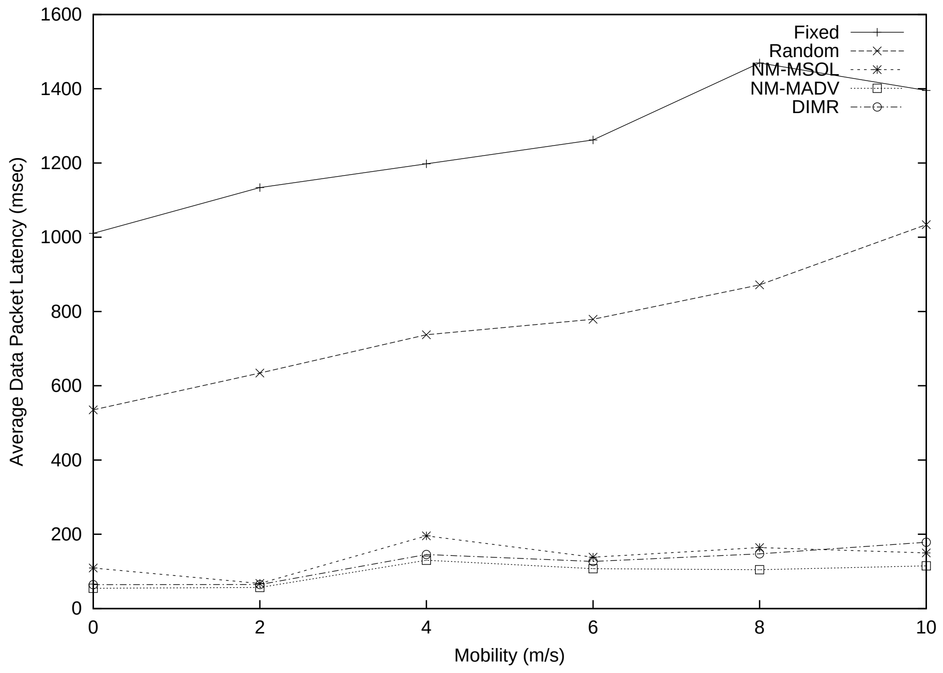

Fig. 16. Packet latency vs. mobility

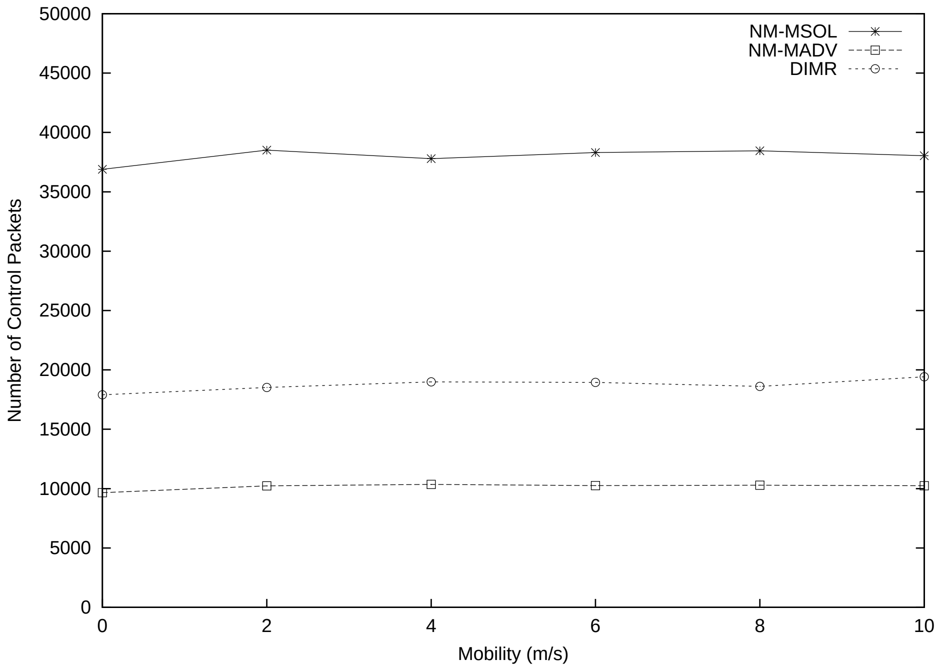

Fig. 17. Total overhead vs. mobility

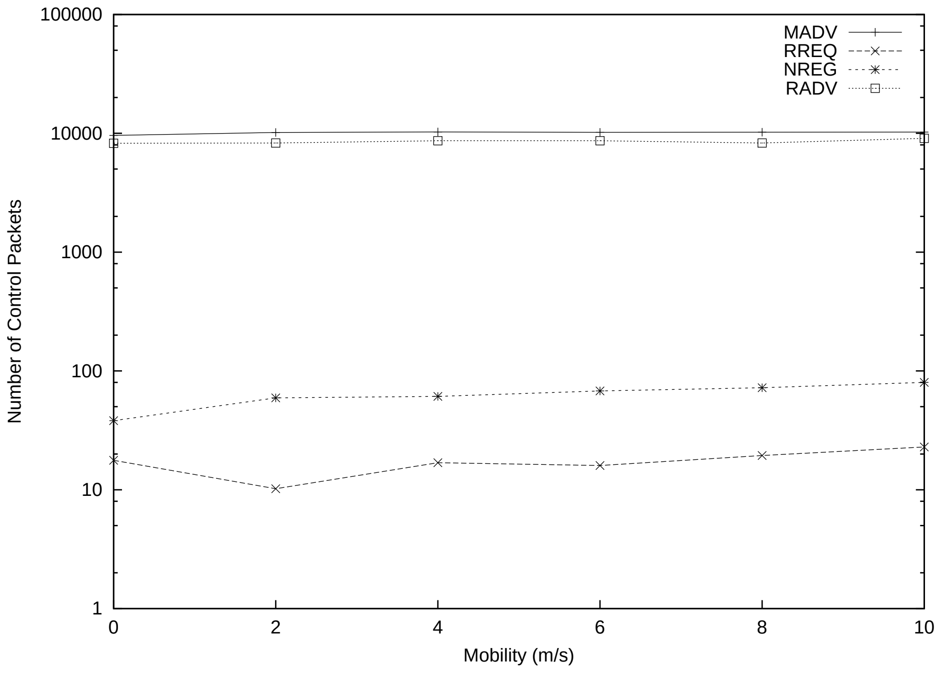

Fig. 18. Distribution of overhead in DIMR vs. mobility

2. Simulation Results

Figs. 15-18 compare the performances of all MIX route algorithms as node mobility varies from 0 to 20 m/s. The number of connections is 10 and the number of MIX nodes is 5. Each value represents an average of 60 simulation runs. From Fig. 15 and Fig. 16, we can see that all adaptive algorithms show significantly better performance than two non-adaptive algorithms. Fig. 17 shows that the number of control packets generated by each adaptive algorithm does not increase as node mobility increases. This explains why the algorithm is not sensitive to node mobility. In the figure, it is shown that MIX solicitation generates more control packets than MIX advertisement and DIMR. The reason is that each MSOL packet is flooded to all nodes within a network while each MADV packet has a limited propagation range. Although each RADV packet is also flooded to the entire network, the number of MIX nodes is typical lower than that of mobile nodes and the route advertisement interval is larger than MIX solicitation interval in our simulations. Fig. 18 shows the distribution of control packets generated by the DIMR algorithm. Most of control packets are MADV and RADV packets. RREQ and NREG packets only make up a negligible portion of the total overhead.

Figs. 19-22 compare the performances of all MIX route algorithms, with different numbers of MIX nodes. The number of connections is 10 and the node mobility is 8 m/s. Each value represents an average of 60 simulation runs. We also calculate a 95% confidence interval for each mean. Since the confidence intervals with respect to three adaptive algorithms are quite small, we do not plot them in the figures. From the simulation results, we can see that, when there is only one MIX, all algorithms yield similar performance. As the number of MIXes increases, the performances of two non-adaptive algorithms are not stable and poor, which is demonstrated by the large

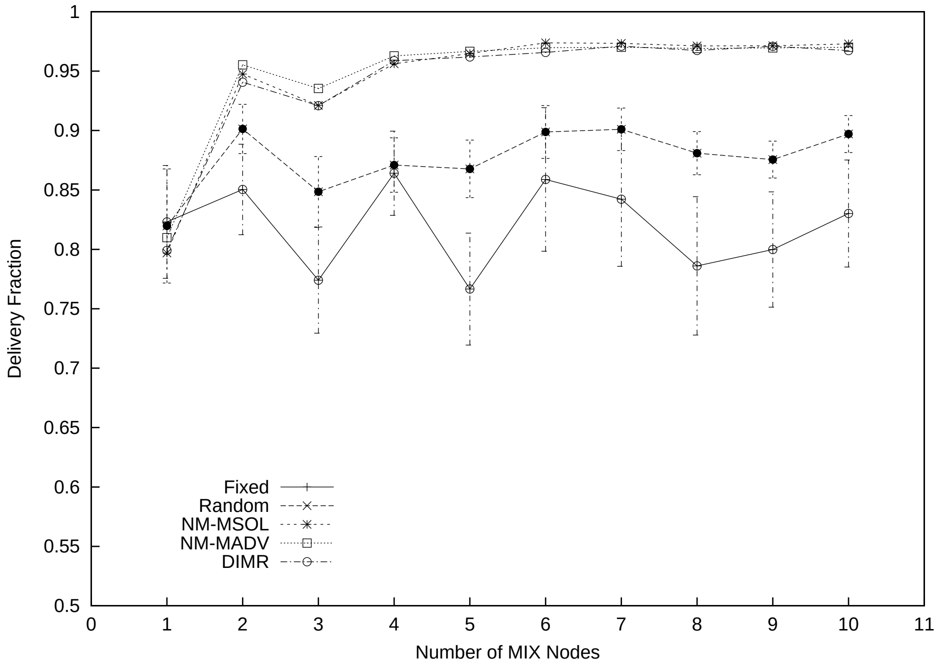

Fig. 19. Delivery fraction vs. number of MIXes

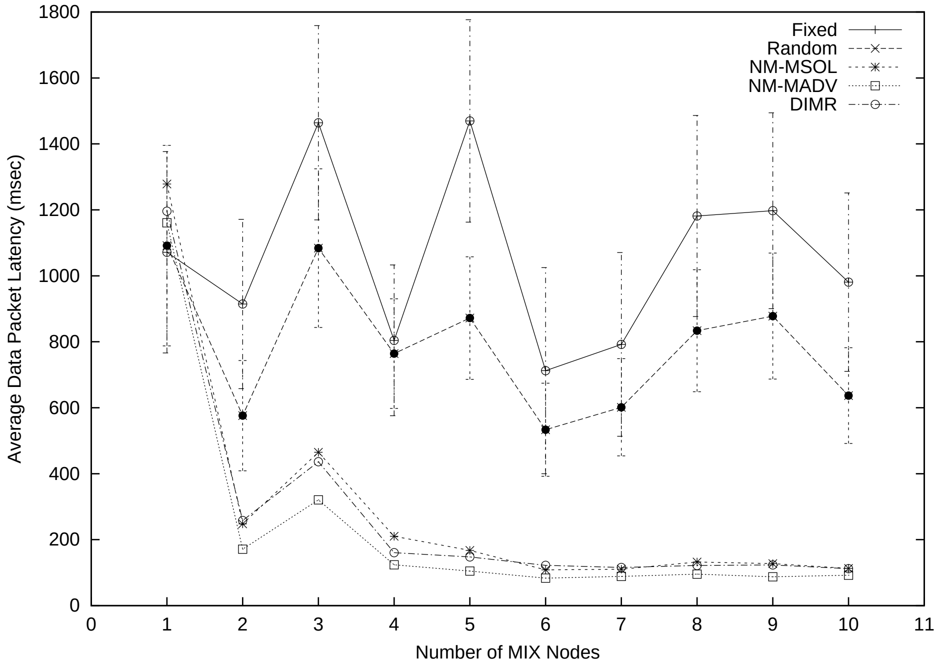

Fig. 20. Packet latency vs. number of MIXes

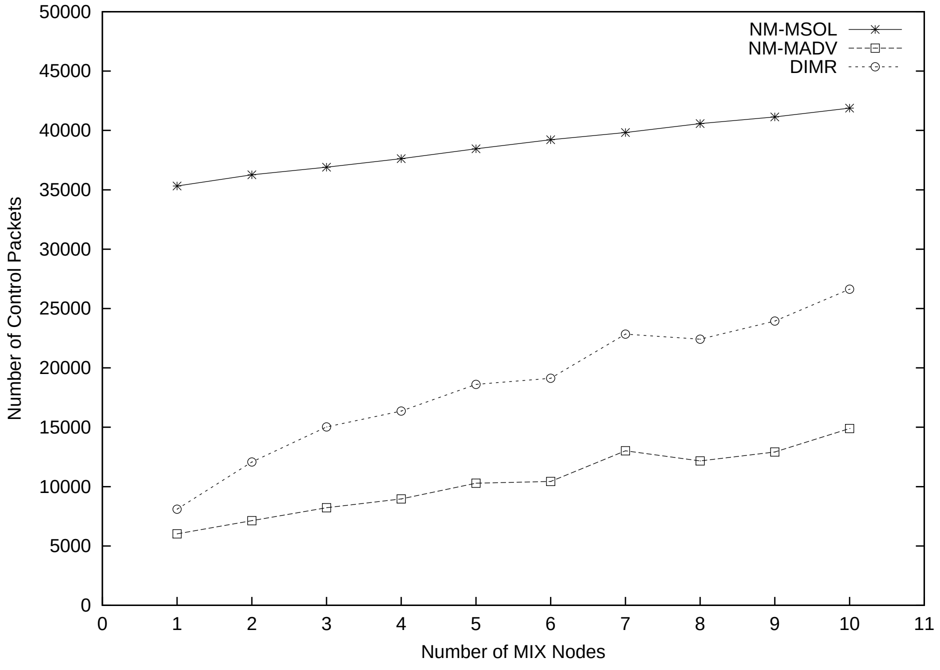

Fig. 21. Total overhead vs. number of MIXes

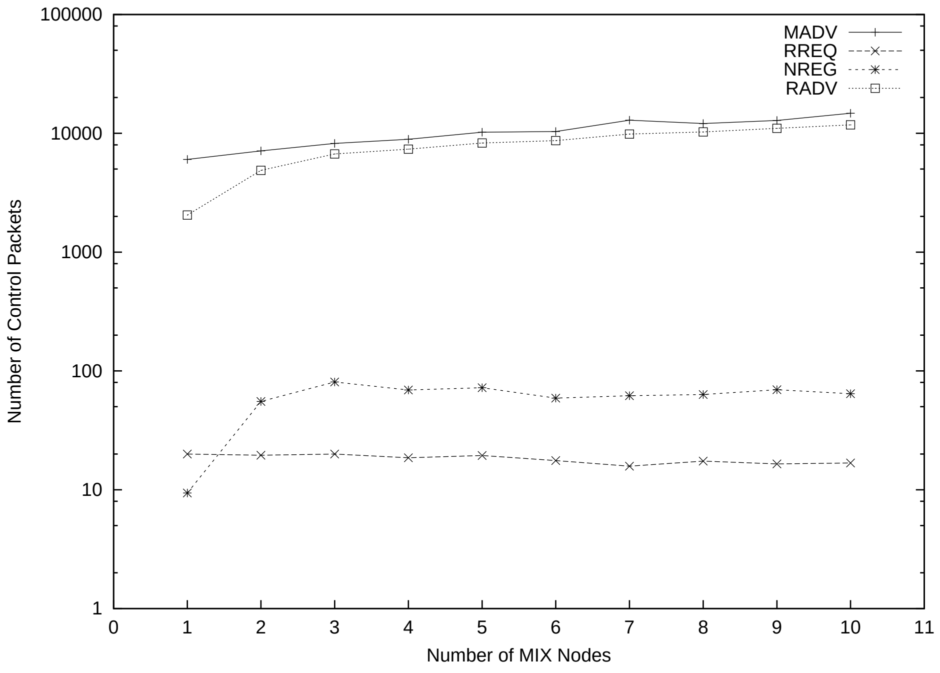

Fig. 22. Distribution of overhead in DIMR vs. number of MIXes

confidence intervals, whereas all adaptive algorithms show stable and much better performance. Fig. 21 shows that the control overheads of all adaptive algorithms increase as the number of MIX nodes increases, and the DIMR algorithm demonstrates a higher speed than other two algorithms. In section 3 and 1, we pointed out that the NM-MADV algorithm overhead and the DIMR algorithm overhead are $O(nm)$ and $O((n+m)m)$ respectively. The latter is obviously more sensitive to increase of the number of MIX nodes. Simulation results confirm this observation. Fig. 22 shows the distribution of control packets generated the DIMR algorithm. As expected, MADV and RADV packets make up the majority of the overhead.

F. Related Work

The MIX-net concept proposed by Chaum [14] was first applied to e-mail applications to provide anonymity support. Anonymous remailers in the Internet are loosely classified into four types. Type 0 remailer simply strips headers and resends the contents to the destination [95]. This leaves a single point of compromise and failure in anonymity service. In 1996, legal pressure forced the operator to reveal actual sender-recipient correspondence. Type 1 remailer (or "Cypherpunk") uses a set of remailers equipped with some MIX features [43]. The mail sender selects a chain of remailers, then forms an onion-like message using the public keys obtained by PGP. Each remailer could only see the address of the next remailer. Type 2 remailer (or "Mixmaster") adds more MIX features to Type 1 [20]. An E-mail message is fragmented into a number of fixed-size packets. Each packet is delivered across the chain of remailers and finally reassembled by the last remailer. Thus the system is less vulnerable to content correlation analysis. The extra cost is that new fragment/assemble software is needed in the remailer-net while anonymous E-mails can be delivered in a straight-forward manner in the previous Type 1 system. Type 3 remailer (or “Mixminion”) uses a complex “swap” operation to defend tagging attack [22]. It also requires a single-use “reply block” [33] that is used in anonymous email reply. All four types of remailers assume a fixed MIX-net.

There are many applications that require bidirectional, low-latency communication service. For example, Web browsing, remote login, VoIP. Providing anonymity support to these applications is more challenging than to delay-tolerant message delivery applications. Reed, Syverson and Goldschlag at the U.S. Naval Research Laboratory developed a system that provides anonymous connection services to a wide variety of unmodified Internet applications by means of proxies. The system is called *Onion Routing*. It is composed by a set of *onion routers* which maintain permanent encrypted TCP connections to each other. Anonymous connections are multiplexed over the longstanding connections. For any anonymous connection, the sequence of onion routers in a route is strictly defined at connection setup. Each onion router can only identify the previous and the next hops along the route. Data passed along the anonymous connection appears different at each onion router, so data cannot be tracked en route, and compromised onion routers cannot cooperate by correlating the data stream each sees. Onion routing is different from anonymous remailers in two ways: Communication is real-time and bidirectional, and the anonymous connections are application independent. Onion routing’s anonymous connections can support anonymous mail as well as other applications (e.g., Web browsing).

A large amount of research work has been carried out to find vulnerabilities of the MIX protocol. In [71], Raymond gives a list of known attacks against the MIX back in 2001. Since then, several new attacks are found by researchers. Serjantov and Sewell [77] describe a packet counting attack, where an attacker counts the number of packets that arrive at a MIX and leave the MIX during a certain time interval. Based on the information, the attacker has a high probability of detecting *lone* connections or starting of new connections that pass through the MIX. This attack will be less effective when a connection is not alone on the monitored link. Zhu et al. [103] describe a flow correlation attack, where an attacker analyzes the traffic patterns on all input links and output links of a MIX and attempts to find correlations. The authors consider both time-domain methods and frequency-domain methods for correlation analysis. Fu et al. [29] describe a flow marking attack which targets to MIX nodes in wireless networks. In such an attack, an adversary uses electromagnetic interference to embed a periodic pattern of marks into traffic flows entering a MIX and searches for these marks among traffic flows leaving the same MIX. By tracking these marks, the adversary can discover the communication relationships between users. To detect the pattern of marks, the adversary can use frequency analysis methods which convert the abstract pattern of marks in the time domain to easily detectable invariant frequency components in the frequency domain.

CHAPTER IV

AN ANONYMOUS MAC PROTOCOL

In this chapter, we propose a new design of anonymity system in MANET, which is different from the "classic" hierarchical system in Chapter III. It is based on the peer-to-peer (P2P) concept explored by other researchers [72, 27]. A hierarchical system has inherent limitation in performance due to the fact that a small number of MIX "servers" provide services to a large number of clients. Even though we can allocate more resources (e.g., bandwidth, power) to MIXes, the mobile nodes in a MIX's neighborhood are also under heavy traffic loads and may exhaust resources earlier, which would effectively block access to the MIXes. A P2P anonymity system does not have this problem, because each node becomes a MIX. It also has better security property because an adversary considering active attack now confronts with a much larger MIX set than in a hierarchical system.

Among existing P2P anonymity schemes, Crowds [72] and Tarzan [72] are targeted to Internet users and anonymous Web browsing application. In the designs of both schemes, the MIX [1] routing algorithm is a main component, and actually, the authors' most important contributions. Both schemes use random route algorithm, but Crowds requires a centralized server for storing the MIX pool information while Tarzan provides a distributed "peer discovery" algorithm. Our scheme is fundamentally different in that we focus on anonymous data forwarding, instead of anonymous routing, and provide a MAC-layer solution. Specifically, we design an IEEE 802.11 based MAC protocol that implements anonymous transmission. Our contributions are two folds. First, we show that the broadcast characteristic of wireless transmission

[1]Although the designer of Crowds and Tarzan use different names, we still use "MIX" to refer to a node that relays data traffic for anonymity purpose.

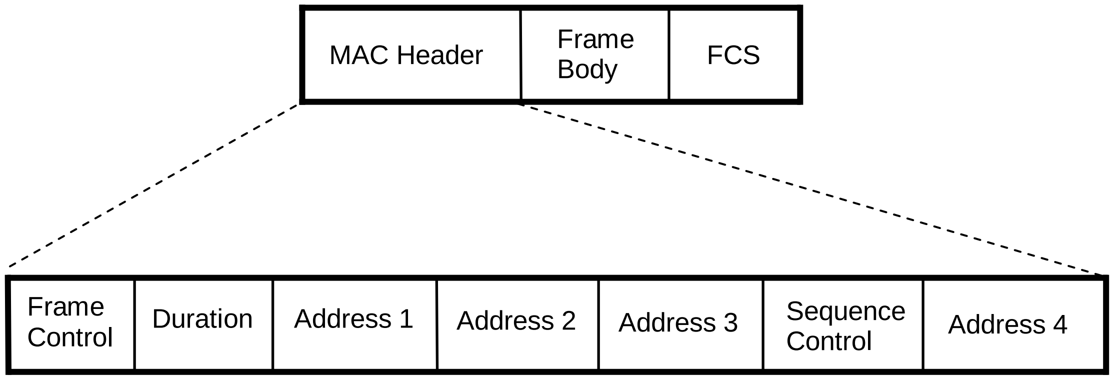

Fig. 23. General format of MAC frame in 802.11 standard

can be utilized to enhance the mixing function. Second, we propose a batched approach to improve reliability of broadcast transmission without sacrificing anonymity.

The rest of the chapter is organized as follows. In section A, we introduce the basic techniques used in our scheme and existing problem. To overcome the problem, we present the design of an anonymous reliable broadcast scheme as well as security analysis of it in section B. In section C, we discuss some issues related to the scheme such as location privacy, sender anonymity, routing protocol, etc. In section D, we present numerical results of traceability analysis and performance evaluation. In section E, we introduce related works.

A. Basic Model and Problem Definition

In IEEE 802.11 network, the unit of transmission at the link layer is *frame*. Fig. 23 shows the general format of a MAC frame. Each frame consists of the following basic components:

a) A *MAC header*, which comprises frame control, duration, address, and sequence control information;

b) A variable length *frame body*, which contains information specific to the frame

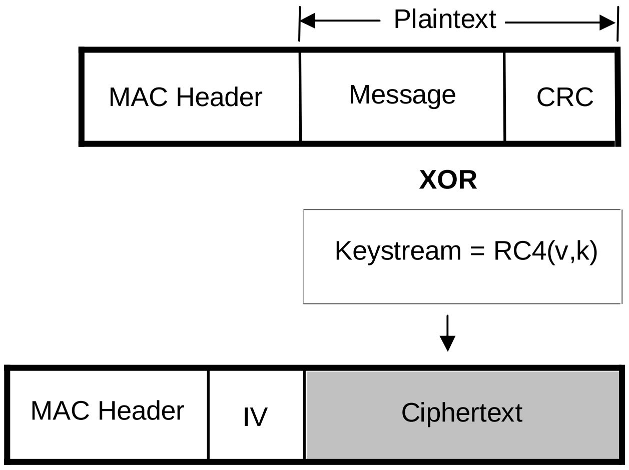

Fig. 24. Encrypted WEP frame

type;

c) A *frame check sequence*(FCS), which contains the IEEE 32-bit cyclic redundancy code (CRC).

In addition to data frame, which carries application data, the protocol uses a number of control frames (i.e., RTS, CTS, ACK) during the operation. The frame type is specified in the frame control field. Depending on the type, the fields Address 2, Address 3, Sequence Control, Address 4 and Frame Body may be omitted. In any data frame, there must be addresses of source and destination node of the frame. If the destination address is all-1's, then it is a broadcast frame; otherwise, it is a unicast frame.

The 802.11 standard defines a Wired Equivalent Privacy (WEP) protocol for encrypting the contents of a data frame, using the RC4 algorithm [44]. WEP relies on a secret key k shared between the communicating parties, i.e., sender and intended receiver of the frame. The operation of WEP is as follows:

1. The sender calculates the CRC of the frame body and appends to it to obtain

a plaintext.

2. The sender chooses an initialization vector (IV) v and uses it as input to generate a *keystream*, i.e., a sequence of pseudorandom bytes equal to the length of the plaintext. The shared key k is another input to the keystream generator.

3. The sender XORs the keystream with the plaintext to obtain the ciphertext.

4. The sender concatenates the IV and the ciphertext to form the new frame payload and transmits it over the radio link (Fig. 24). In order to notify the receiver that it is an encrypted frame, an "encrypted" bit is set in the frame control field.

5. The receiver simply reverses the encryption process to decrypt an encrypted frame. Specifically, it extracts the IV from the frame, generates the same keystream that the sender generated, and XORs it against the ciphertext to recover the initial plaintext.

6. The receiver then checks the CRC on the decrypted plaintext and verifies that the data is not corrupted.

Although serious flaws were found in current implementations of WEP protocol, which may lead to successful attacks against content privacy of encrypted messages, all these flaws can be fixed with improved key management, increased key length, or even new and stronger cryptographic algorithm. So, for the purpose of this paper, we assume that an eavesdropper does not have the capability of performing effective cryptanalysis to break the cipher.

In a wireless network, every transmission is broadcast in nature, regardless of whether it is a broadcast frame or unicast frame, which means that, if there is no interference, every node within the transmission range of the transmitter will receive the frame. It is easy to see that this feature, combined with frame encryption, can be employed to hide the receiver of a unicast frame. The idea is as follows:

1. The sender inserts a pseudo header between the MAC header and the frame body, which comprises the address of the intended receiver of the frame and other control fields. The pseudo header and the old frame body form a new frame body, which is passed to the RC4 cipher for WEP encryption.

2. The sender sets the destination address in the MAC header to all-1's, making it an unsuspicious broadcast frame and transmits it on the radio link.

3. All nodes that receive the transmission will try to decrypt the frame payload. Only the intended receiver can recover the original frame successfully. For other nodes, the CRC verification would fail, which leads to the frame being dropped. The receiver address in the pseudo header should match with the node's address, which gives a double confirmation. After removing the pseudo header, the node can pass the message in the old frame body to the upper layer protocol for further processing.

Although this scheme is easy to implement and provides cheap receiver anonymity, it has a serious reliability problem. In IEEE 802.11 protocol, unicast frames and broadcast frames are treated differently. As we know, wireless transmission often suffers loss due to collision and interference. Upon receiving a unicast frame, the recipient node is required to send an ACK frame to the sender node. If the sender node does not receive the expected ACK after a timeout, it assumes that the previous transmission failed and will retransmit the frame (up to a maximum number) after a random "backoff delay". With stop-and-wait ARQ [86], the MAC protocol can achieve a very high success rate with unicast frame transmission. On the other hand, broadcast frames are transmitted with a much lower reliability in 802.11. The recipient nodes of a broadcast frame are not required to send acknowledgements back to the sender and the sender node does not retransmit a broadcast frame even if it is corrupted or lost during the transmission. Apparently, if no extra measures are taken, the anonymous frames we described above, which are unicast frames originally but converted into broadcast frames, will suffer the low reliability as well. There are several possible solutions to this problem. An intuitive one is to require the intended receiver of an anonymous data frame to send (or broadcast) an anonymous ACK frame after it receives the frame successfully. This approach sounds interesting, but is not secure enough because there exists a timing link between an ACK frame and the previous data frame. So it is subject to traffic analysis attack from eavesdroppers. Another approach is to use existing reliable broadcast schemes. In literature, there are several MAC protocols which were proposed by researchers to improve reliability of broadcast transmission in IEEE 802.11 networks [91, 87, 88, 84, 79, 80]. However, a reliable broadcast scheme has different goal from an anonymous transmission scheme. In the former, a sender wishes that all neighbors receive a transmitted frame. To achieve the goal, the frame may need to be retransmitted for many times until either all neighbors receive it (with ACKs) or the sender gives up. If we use this scheme to provide anonymity, the utilization of the system could be low, because a lot of retransmissions are unnecessary.

In the next section, we present a new approach to reliable anonymous transmission of unicast frames. This approach is based on batching technique to implement anonymous data transmission and polling technique to implement anonymous acknowledgment. Reliability is achieved by retransmitting lost frames. So it requires the receiver of an anonymous data frame to acknowledge receipt. However, instead of the receiver node sending an ACK frame actively, the source node sends POLLs to a set of neighbor nodes and receives REPLY's from each of them. The set of polled nodes serves as an anonymity set for the receiver node.

B. A Reliable Anonymous Transmission Scheme

1. Preliminary

IEEE 802.11 MAC protocol uses the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) scheme for basic channel access. It works as follows. A node wishing to transmit senses the channel. If the channel is free for a time equal to the $DIFS$ interval (50 μs in our simulations), the node transmits. If the channel is busy, the node enters a state of collision avoidance and backs off from transmitting for x slots of time, where x is a random number within the contention window. In the collision avoidance state, the node sensing the channel busy will suspend its backoff timer, only resuming the backoff countdown when the channel is again sensed free for a $DIFS$ period. The carrier sensing range R_{cs} determines how far a node's transmission can be detected. It is decided by the antenna sensitivity and usually fixed (550 $meters$ in our simulations).

When a node transmits a frame, the signal is propagated from the transmitter to a receiver. During the propagation, the signal suffers attenuation and loses considerable power. When a signal arrives, whether a receiver can decode it and receive the transmitted frame correctly or not depends on two conditions:

1. The receiving signal power is above the $Rx\_THRESHOLD$;

2. The signal-to-noise ratio (SNR) is above the $CAPTURE\_THRESHOLD$.

The $Rx\_THRESHOLD$ decides the transmission range R_{tx}, within which a

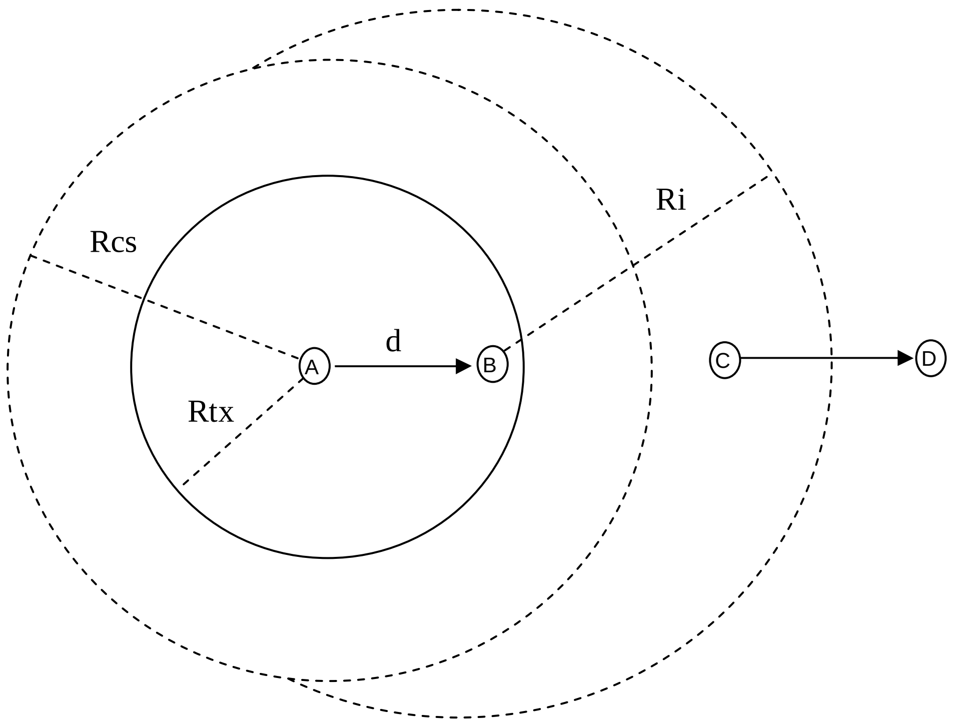

Fig. 25. Illustration of hidden node problem

frame can be successfully received if there is no interference from other radios. The *CAPTURE_THRESHOLD* decides the interference range R_i, which is the maximum distance between the receiver and an effective interfering node. Both R_{tx} and R_i are determined by transmitting power, antenna gains, and signal attenuation model. In addition, R_i is a function of distance between transmitter and receiver, d, as well. According to [97], when signal attenuation follows the two-way ground model, the following relationship exists:

$$R_i = 1.78d \tag{4.1}$$

From equation (4.1), a larger d corresponds with a larger interference range, which means a higher probability that a transmission experiences interference. When d is larger than $R_{cs}/2.78 = 0.36 \times R_{cs}$ (198 *meters* in our simulation), there might exist nodes within the interference range of the receiver, but beyond the carrier sensing range of the sender. These nodes are called *hidden nodes*. When a hidden node is transmitting, the sender cannot sense it and will start its own transmission, but the transmission will fail because the hidden node's transmission interferes with it. This problem is illustrated in Fig. 25, where node C is a hidden node to node A. In this example, transmission from C to D can go through and is not interfered by A's transmission.

Hidden nodes are very harmful to the system performance. So IEEE 802.11 protocol provides an optional RTS/CTS handshake to deal with the problem. Before transmitting a data frame, the sender can send a short RTS frame, which is replied by a CTS frame from the recipient, to reserve channel. The duration field in RTS and CTS specifies the length of time interval that the future data transmission endures. Any node hearing the RTS or CTS should remain quiet until the reserved transmission is finished. This is also called *virtual carrier sensing*. Specifically, each node maintains a NAV (Network Allocation Vector) which remembers the remaining time of an ongoing transmission. The RTS/CTS handshake scheme only partially solves the hidden node problem. As demonstrated in [97], it is not very effective when transmitter-receiver distance is relatively large, because RTS and CTS can only be received by the sender's and the recipient's neighbors, i.e., nodes within each's transmission range R_{tx}, but not all potential interfering nodes.

When we design the anonymous MAC protocol, we use the physical carrier sensing as main method for suppressing competing transmissions. Due to anonymity requirement, each node (sender) must transmit data frames in batches and the data frames in each batch can be addressed to different nodes (receiver) within the transmission range (250 $meters$ in our simulations). To increase the probability of successful transmission, the sender uses a polling mechanism to check the availability of each receiver before transmitting data. During the polling process, the sender sends polls to each receiver individually. If a polled node successfully receives the poll and replies, then it means that there is no interference at the node and the node is available for receiving data frames. Polls and replies are transmitted alternatively with a small time interval between them. Note that the maximum distance between any two neighbors of the sender (500 $meters$) is smaller than the carrier sensing range R_{cs} (550 $meters$). Thus, each node within the sender's transmission range will sense a busy channel during the polling process and will not try to transmit. On the other hand, the maximum distance between the sender and any node within a receiver's transmission range (500 $meters$) is also smaller than the carrier sensing range R_{cs}. Thus, the sender's transmission of polls will suppress those nodes' attempt of transmitting as well. However, like RTS/CTS, our scheme does not prevent a hidden node from disrupting polling and transmission of data frames afterwards. In this case, the sender executes a binary exponential backoff algorithm and retry after a random number of time slots.

2. Protocol Description

We first give a brief description of the proposed scheme. The details of the design are elaborated in the following sections.

- In order to hide the receiver, each unicast frame is transformed into an encrypted broadcast frame, as described in Section A, with receiver address being saved in the pseudo-header and being hidden. To an outside eavesdropper, all transmissions appear as broadcast frames.
- Each node transmits anonymous data frames in batches. The frames in a batch are addressed to different nodes within the transmission range, i.e. neighbors.

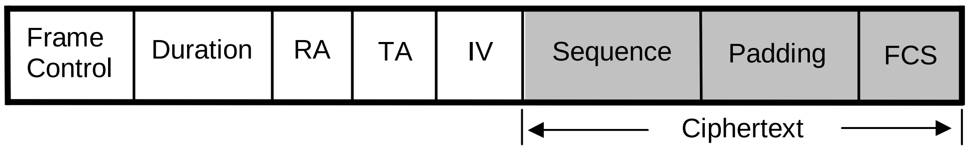

Fig. 26. POLL frame format

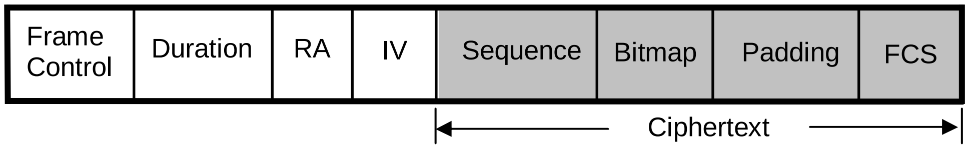

Fig. 27. REPLY frame format

Instead of sending frames blindly, a sender checks the availability of particular neighbors at first by means of polling. Specifically, the sender transmits POLL frames to a set of neighbors one by one and expects to receive REPLY frames from each of them. If a polled node does not reply, then it means that the node did not receive the POLL frame, probably due to interference, and thus, is not available for receiving data frames thereafter. Based on the replies, the sender constructs a batch of data frames and sends all out. To increase the probability of successful reception, data frames addressed to neighbors that reply polls have higher priority of being selected into the batch.

- In addition to interference sensing, the POLL/REPLY frames have another functionality, i.e., acknowledgement. Each frame is assigned a sequence number. In a POLL, the sender queries about receiving status of transmitted frames. In a REPLY, the polled node reports the sequence numbers of received frames. We use a Selective Repeat ARQ protocol [86]. The sender only retransmits lost frames (up to a maximum number of retransmissions).

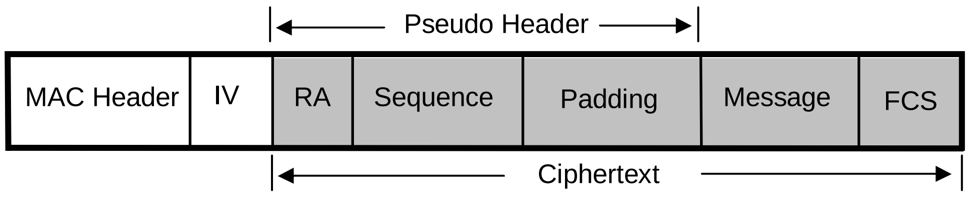

Fig. 28. Anonymous data frame

a. Frame Format

Fig. 26 shows the format of a POLL frame, where *RA* is the address of the node being polled, TA is the address of the node transmitting the POLL frame, *Duration* is the length of time required to complete the current poll, which is calculated as the transmission time of a REPLY frame plus one $SIFS$ interval (20 μs in our simulations), *Sequence* is used by the ARQ protocol, and *Padding* is a number of random bytes. The last three fields in a POLL frame are encrypted.

Fig. 27 shows the format of a REPLY frame, where *RA* is the address of the node transmitting POLL, *Sequence* and *Bitmap* are used by the ARQ protocol, and *Padding* is a number of random bytes. The last four fields in a REPLY frame are encrypted.

Fig. 28 shows the format of an anonymous data frame, where *RA* is the address of the intended recipient node, *Sequence* is the identification number of the frame, and *Padding* is a number of random bytes. The three fields above comprise the pseudo header.

b. Sender's Protocol

Each node maintains a FIFO queue, holding frames that are waiting to be transmitted or retransmitted. When a new frame is received from the upper layer, it is given a sequence number. The sender and receiver use this sequence number to track and retransmit lost frames. For this purpose, each node i maintains a variable SN_{ij} with respect to each neighbor node j. SN_{ij} is initiated to 0 at the system setup time. When there is a new frame to node j, node i assigns the current value of SN_{ij} to the frame and increments SN_{ij} by 1. This ensures that node j receives frames from node i with contiguous sequence numbers. If a number is missing, the frame must be lost during transmission.

At each node i, with respect to each neighbor node j, a sending window $[LSN_{ij}, HSN_{ij}]$ is maintained to record the range of sequence numbers of frames stored in the queue. LSN_{ij} is the lowest sequence number of frames, from i to j, currently in the queue, while HSN_{ij} is the highest sequence number. Node i advances LSN_{ij} in two cases:

a) Node j acknowledges receiving of the frame with sequence number LSN_{ij};

b) Node i fails to transmit the frame with sequence number LSN_{ij} after a maximum number of attempts (4 in our simulations) and discards it. For this purpose, node i maintains a "retry counter" for each frame in the queue, recording the number of retransmissions.

At each node i, if the queue is not empty, the following algorithm is executed:

1. Node i executes the carrier sensing and collision avoidance procedure.

2. Node i constructs a polling set by adding all receivers of frames currently in the queue. If the polling set size is smaller than a preset minimum value (referred to as $MIN\_POLLING\_SET\_SIZE$), nodes in i's neighbor set are randomly selected to add in.

3. Node i polls each node in the polling set at a random order. It proceeds as follows. Node i sends a POLL frame to the first node and listens. If the polled

node is j, then the sequence field in the POLL frame carries the current value of LSN_{ij}. If the channel is still free after two $SIFS$ intervals, the node i polls the next node. Otherwise, it prepares to receive the REPLY frame transmitted by the polled node. If a valid REPLY frame is received, node i will update its state based on information in it (e.g., releasing acknowledged frames, advancing the sending window, incrementing retry counters of unacknowledged frames), and sends the POLL frame for the next node after one $SIFS$ interval. If node i receives a corrupted REPLY, probably due to interference, or if the channel is sensed busy during the SIFS interval, node i will abort the current transmission and go to step 1 after backing off a random number of slot times. If a node fails to reply consecutive pollings for a maximum number of times (7 in our simulations), the link is assumed to be broken and all frames to be sent on that link are purged from the sender's queue. The retry limit for polling is set higher than that for retransmitting data. This prevents unnecessary loss of data when the receiver is just experiencing transient interference.

4. If node i reaches this step, there is only one possibility, where all nodes in the polling set have been polled and node i either receives a valid REPLY or nothing from each node. The nodes from which REPLY frames are successfully received are so-called "available receivers". If the set of available receivers is empty, then node i will abort the current transmission and go to step 1 after backing off a random number of slot times; otherwise, node i selects a set of data frames (referred to as *batch*) from its sending buffer to transmit. The batch size is controlled by two system parameters: MIN_BATCH_SIZE and MAX_BATCH_SIZE. When node i constructs the batch, it should choose frames addressed to available receivers first. If all such frames have been chosen and the batch size

is still less than MIN_BATCH_SIZE, then node i can choose frames addressed to other receivers. In our experiments, MIN_BATCH_SIZE and MAX_BATCH_SIZE take values of 1 and 4 respectively. During transmission, the frames in a batch are transmitted consecutively with a time spacing equal to $SIFS$.

c. Receiver's Protocol

At each node j, with respect to each neighbor node i, a receiving window is maintained to record the sequence numbers of received frames. In Selective Repeat ARQ protocol, a common approach is to use two variables to implement a receiving window: a Lowest Bound LB_{ji} and a one-byte Bitmap BM_{ji}. All frames from i with sequence numbers lower than LB_{ji} have been received. The BM_{ji} indicates the receiving status of frames whose sequence numbers higher than LB_{ji}. Specifically, if the k-th bit of BM_{ji} is 1, it means that the frame with sequence number $LB_{ji}+k$ has been received. For example, a LB_{ji} of 100 and a BM_{ji} of 11100110 indicate that node j has correctly received frames 0-99, 101, 102, 105, 106, 107, whereas frames 100, 103, 104 were lost. Node j advances its receiving window in two cases:

a) When a POLL from node i is received, if $LSN_{ij} > LB_{ji}$, it means that the sender node i has advanced its sending window and given up its attempts to retransmit frames lower than LSN_{ij}. This could happen when node j experienced temporary severe interference and failed to reply node i's pollings for a certain number of times. In this case, node j synchronizes its receiving window with node i's sending window by advancing LB_{ji} to LSN_{ij}.

b) When a data frame from node i is received, if its sequence number matches with LB_{ji}, then node j can advance its receiving window, i.e., incrementing the LB_{ji}

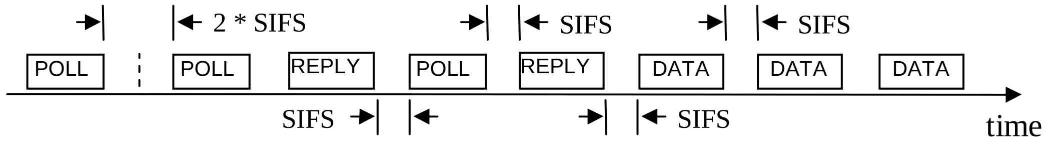

Fig. 29. An illustration of the scheme

by 1 and right-shifting the BM_{ji} for one bit. Node j can repeat the adjustment until the lowest bit of BM_{ji} is 0. If the sequence number of the received data frame is larger than LB_{ji} and is not a duplicate, the BM_{ji} is updated to indicate the receiving status.

Unlike many Selective Repeat ARQ based protocols, we do not maintain a "receiver buffer" at the MAC layer to hold out-of-sequence frames. Instead, a receiver passes each received frame immediately to the upper layer (i.e., network). There are two reasons. First, this reduces the queueing delay. Second, frames transmitted on a link belong to different end-to-end flows and typically have different next hop receivers. Frame loss of one flow should not affect the frame delivery of other flows. This is similar to the head-of-line problem in router design. By relaxing the in-sequence constraint, we can increase the overall network throughput. Notice that to provide reliable message delivery for users, the destination node now has responsibility for sequencing.

The described protocol is illustrated in Fig. 29. In the figure, the first polled node does not send a REPLY frame, probably not receiving the POLL. Therefore, the sender sends the second POLL (to a different node) after two $SIFS$ intervals. Since any node can transmit if the channel remains free for $DIFS$, having sender transmitting the second POLL earlier, without waiting for the transmission time of a REPLY frame, prevents any neighbor from interrupting the polling process. The second and third POLLs are replied. Each polled node transmits the REPLY frame immediately, after one $SIFS$ interval. Data frames in the current batch are transmitted continuously, with one $SIFS$ spacing between two consecutive frames. So, during the entire process, the medium is never idle for more than $2 \times SIFS$.

3. Security Analysis

In this section, we analyze how this scheme is effective in hiding the recipient of a transmitted data frame. For this purpose, we discuss attacks that an adversary can perform to reveal the information hidden in the encrypted pseudo header. We do not consider cryptanalysis attack, because if the adversary is so powerful to be able to break encryption, there is no any scheme that can provide sound security. However, the risk of a node being compromised by the adversary does exist, especially in hostile environment. Similarly, we assume that the proportion of compromised nodes is not too high.

a. Node Intrusion

If a node is compromised, the adversary can know whether a received frame is for the node or not. It can use this information to reduce the scope of possible receivers of the frame, i.e., anonymity set. In our scheme, the receiver anonymity set of a frame is not the set of all nodes within the sender's transmission range, but a subset of it, i.e., "available" nodes. This property may help the adversary to identify frame receiver. Notice that we could have adopted a more secure design in which each sender node always polls all its neighbors and transmits data frames according to their orders in the queue (without checking receiver availability). However, our simulation demonstrates that the performance of this design is poor when the average node degree is more than 6. The current design tries to implement a trade-off between security and performance. In section D, we show the effects of different

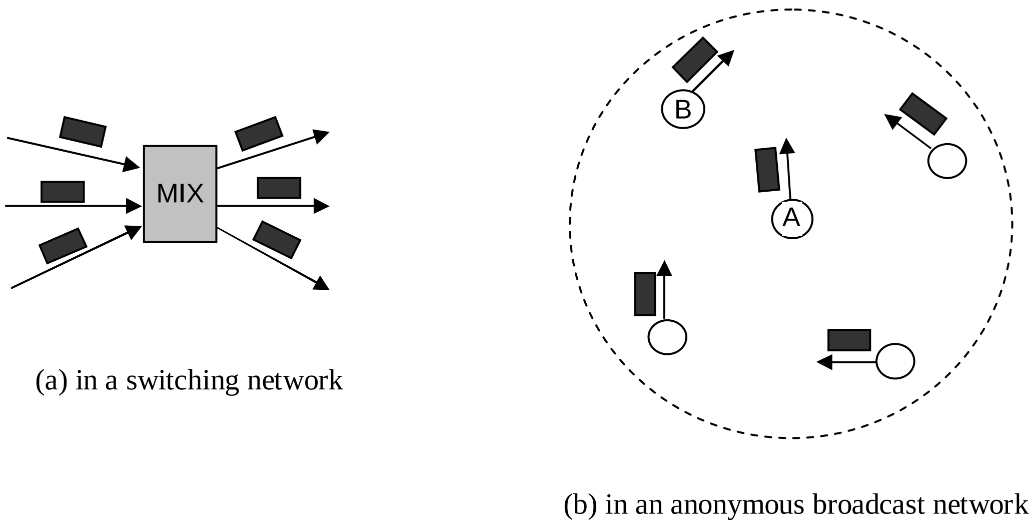
MIX
(a) in a switching network
B
A
(b) in an anonymous broadcast network

Fig. 30. Different attacking scenarios against MIX

$MIN\_POLLING\_SET\_SIZE$ values on security and performance.

b. Traffic Analysis Attack

This represents an attack that can be performed by an external adversary to reveal correlation between two data objects, based on observed characteristics.

For a conventional MIX, the attacker tries to find correlation between an input message and an output message of the MIX. To achieve this goal, the attacker can utilize message content, size, timing information, or can manipulate the input and output messages. Specifically, *content attack* compares the contents of two messages bit by bit, looking for match; *size attack* examines the message lengths and is only effective against protocols using variable-length messages; *timing attack* searches for temporal dependencies between transmissions. *Flooding attack* (also called *node flushing attack* [14], $n-1$ *attack* [76]) is a special form of content attack. In case of a simple threshold n MIX, which flushes after receiving n messages, the attack proceeds as follows: When the attacker observes a targeted message entering the MIX, it sends $n-1$ messages into the MIX to make it fire. In Chaum's design [14] of MIX-net, the sender of a message encrypts it with the MIX's public key while the MIX decrypts the message and retransmits it. So the attacker can recognize his own messages when they leave the MIX. Among n messages leaving the MIX, the one which is unrecognizable must be the targeted message.

The above description of traffic analysis attacks applies to MIXes in a switching network. In an anonymous broadcast network, there is no explicit "link" between two nodes because the destination address of each message is the broadcast address. Thus, the attacker needs to find correlations between apparently independent transmissions by different nodes (see Fig. 30). For example, node A transmits a frame at time t, and node B, one of its neighbors, transmits at time $t+\epsilon$. This may suggest that node B is the receiver of node A's frame and is forwarding the frame to its next hop. However, for this timing attack to succeed, the following conditions must be satisfied:

1. The queue is empty when node B receives the frame, and
2. All other neighbors of node A have no frames to transmit.

If any of the above conditions is not satisfied, then the probability of a successful attack would be reduced, due to a larger delay between two transmissions of the same frame. This suggests that each node having a non-empty queue, i.e., always in saturation mode, has benefits to security. The queue here serves a similar function as the "pool" in a conventional MIX. Again, there is a trade-off between security and performance. In the current design, the scheme does not generate dummy data frames, and only generates dummy polls, based on the assumption that network users provide enough traffic loads. However, it can be easily extended to apply to low-traffic networks, by allowing nodes to generate dummy data frames. It worths noting that the proposed scheme does batching and reordering in a different fashion than a conventional MIX. Frames are transmitted first-in-first-out on a per each destination basis, but on the node level, frames are transmitted in a different order than when they arrive. The scheme is also very efficient in achieving the security goal. With one broadcast, all neighbors receive a masked data frame. To an unintended receiver, it provides a cover for the node's ensuing transmissions. To achieve the same effect in switching network, multiple transmissions on explicit links to neighbors are needed.

In addition to timing attack, the proposed scheme is also resistant to other attacks. To prevent content attack, the ciphertext in a retransmitted frame should be recalculated based on new IV and new padding value. To prevent size attack, all frames should have identical size. Per-hop encryption of frames effectively prevents flooding attack.

C. Discussions

1. Why Not Hide the Sender?

In our proposal, we only hide the receiver of a frame. It is appealing to hide the frame sender as well. At the first glance, this is simple to do. We can move the real source address from MAC header to the encrypted pseudo header and set the apparent source address to all-1's. Or we can use the temporary MAC address scheme proposed by Gruteser and Grunwald [31], in which a node switches its address frequently. However, these schemes are not very effective when an adversary can use other information than MAC address to determine the sender of a frame.

- On the physical layer, it is usually possible to locate a sending device by recording signal delays and performing triangulation. If a node moves infrequently, the adversary can easily determine whether two transmissions are from the same node. Other radio signal properties, such as signal strength, signal-to-noise radio, can also be utilized for this purpose [4, 11, 56, 81].

- In pervasive computing environment, user's current location is an important information that an application uses to tailor its functionality. There are a number ways of collecting location data (e.g., GPS, Active Badge [94], Cricket [68]). Typically, location data is transmitted to and stored at a location server, which is then accessed by external services. If the transfer and storage of location data are not secure enough, an adversary has chance to obtain the information.
- Attacker may also obtain information by configuring devices to observe their environment. The most obvious problem is the deployment of many near-invisible cameras in a monitored area (e.g., battlefield). Significant progress is being made in sensor network technology to realize omnipresent monitoring capability as such. In this situation, user's movements are under complete surveillance. Even if a node suspends transmission, changes position and begins new transmission with a different ID, it can be still be recognized and linked to its old ID.

To summarize, we conclude that sender anonymity is not reliable if there is no location privacy. In this paper, we assumes a strong adversary model. If the adversary does not have the capability we described, the mentioned sender anonymity schemes certainly will enhance the system and make it difficult for the adversary to perform traffic analysis.

2. Is Anonymous Routing Necessary?

Recently, many anonymous routing protocols have been proposed for MANET [55, 96, 9]. The purpose of these protocols is to find paths between source-destination node pairs and set up anonymous connections for data transfer. During the process, all control messages are encrypted and intermediate nodes do not know the initiator, nor the target node, of the messages. Taking ANODR [55] as an example. During the RREQ phase, a RREQ message carrying "trapdoor" information is flooded across the network. The path that the RREQ traverses is recorded in a cryptographic onion embedded in the message. Only the destination node can open the trapdoor and reply a RREP message. During the RREP phase, the RREP message is forwarded along the RREQ path, in reverse, back to the source node. While forwarding the RREP, each intermediate node sets up mapping between two "route pseudonyms", one representing the upstream node and another representing the downstream node, in its connection table. This information will be used during data transfer. It can be seen that, in such a protocol, the well-known route repair and route optimization techniques in the common routing protocols, like DSR, cannot be applied. Our scheme overcomes this problem, based on a different notion.

In our notion, data transfer is separate from routing path setup. Each node can use the DSR protocol to find routing paths, set up anonymous connections like in onion routing, and transmit data using the anonymous MAC protocol. To find a routing path, the source node does not always need to send RREQ. In DSR, a node can learn paths from other nodes' messages (e.g., RREP). Or each node can send dummy RREQ messages to confuse the eavesdroppers. Therefore, a node trying to find paths to certain nodes does not necessarily initiate data communications with those nodes.

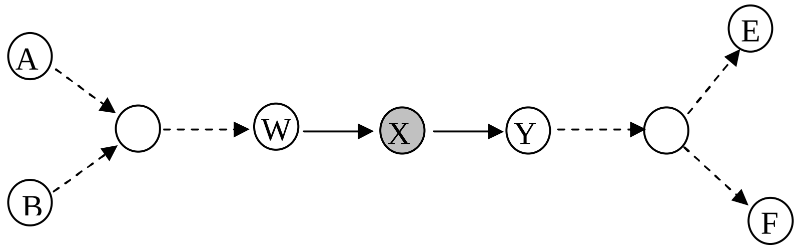

Fig. 31. An illustration of exposed segment

D. Numerical Results

1. Traceability

In this section, we investigate the traceability of an end-to-end connection in presence of compromised nodes. For connection c, the traceability p_c is defined as the probability that the source and destination of the connection be detected by the adversary. If the connection's source or destination is a compromised node, then its traceability is 1. So, in the following, we consider the case that both ends of the connection are trustworthy.

Suppose that a node X is compromised, when it receives a packet from neighbor W and is required to forward the packet to neighbor Y, it immediately knows that the three nodes (W, X, Y) are consecutive on a connection path. We call it an *exposed segment* (see Fig. 31). If either W or Y is compromised, the exposed segment can be extended to W's or Y's neighbor, and so on. The adversary tries to find all connection paths sharing an exposed segment and makes a guess about which connection the packet belongs to. For instance, there are N connections and each has n_i $(i = 1, 2, \cdots, N)$ paths sharing the segment. Then the probability that the packet belongs to the ith connection is $\frac{n_i}{\sum_{j=1,2,\cdots,N} n_j}$.

Assume that each connection c has a path set $P^{(c)}$ and the paths in it are used to deliver packets in a round-robin fashion. Since each path contains different exposed

Table III. Numerical Results for Connection Traceability

compromised nodes (%)	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45	0.5
traceability	0.08	0.2	0.27	0.38	0.44	0.53	0.59	0.67	0.72	0.78

segments, the risk of a packet being exposed on each path is different. When a packet take a path which contains K non-adjacent exposed segments, the adversary has K chances to make a correct guess about which connection the packet is associated with. The probability of successful detection is

$$1 - \prod_{i=1}^{K}(1 - p_i) \tag{4.2}$$

where p_i is the adversary's successful-detection probability at the i-th segment. The traceability of the connection is the average detection probability over all connection paths, since each path has the same chance of being taken.

To get numerical results, we generated a 50 node random network and set up connections between all node pairs. For each connection, we find the shortest path set. Given the percentage of compromised nodes, we chose compromised nodes randomly from the node set. Table III gives the average traceability over all connections as the percentage of compromised nodes varies. It is shown that when having less than 10% of compromised nodes, the connection traceability is less than 20%. When 50% nodes are compromised, most of the connections could be traced.

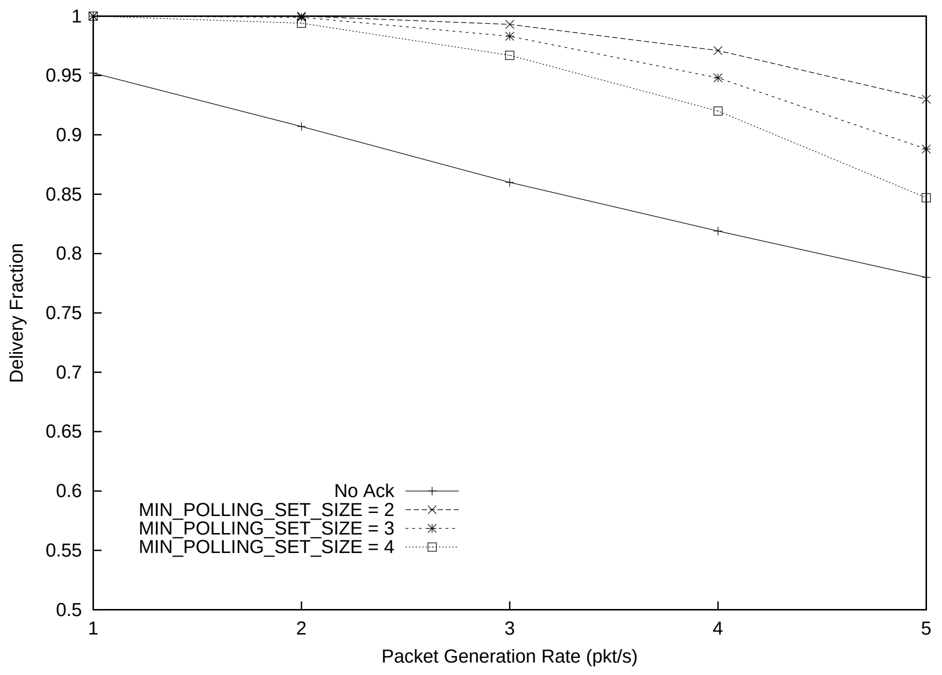

Fig. 32. Data packet delivery ratio

2. Performance

In this section, we present simulation results on the performance of the proposed scheme. We simulated the scheme using the ns-2 simulator [7] and carried out simulations in a 50-node static network. Nodes are randomly distributed in a 1000m x 1000m square area. There are 20 CBR connections in the network with source and destination of each connection being randomly picked. The source node of each connection generates data packets with fixed size, namely 512 bytes. We vary the average data generation rate to produce different traffic loads.

In Fig. 32, we show the data packet delivery fractions under different traffic loads. For comparison purpose, we also show the performance of “pure” broadcast scheme, i.e., without acknowledgment. We can see that even with light traffic load, the pure broadcast cannot ensure delivery of all frames, and when traffic load increases,

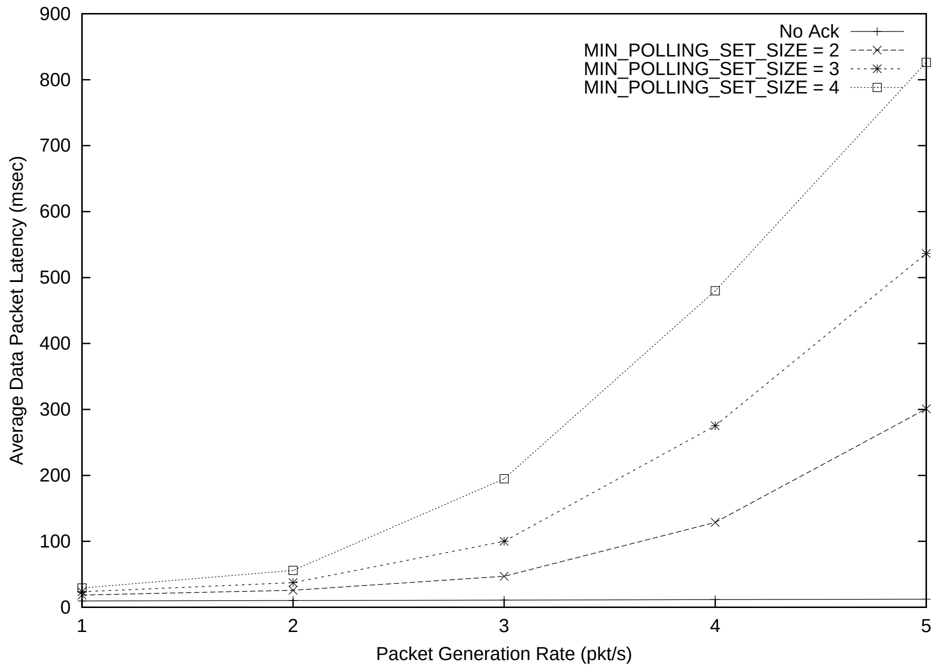

Fig. 33. End-to-end data packet latency

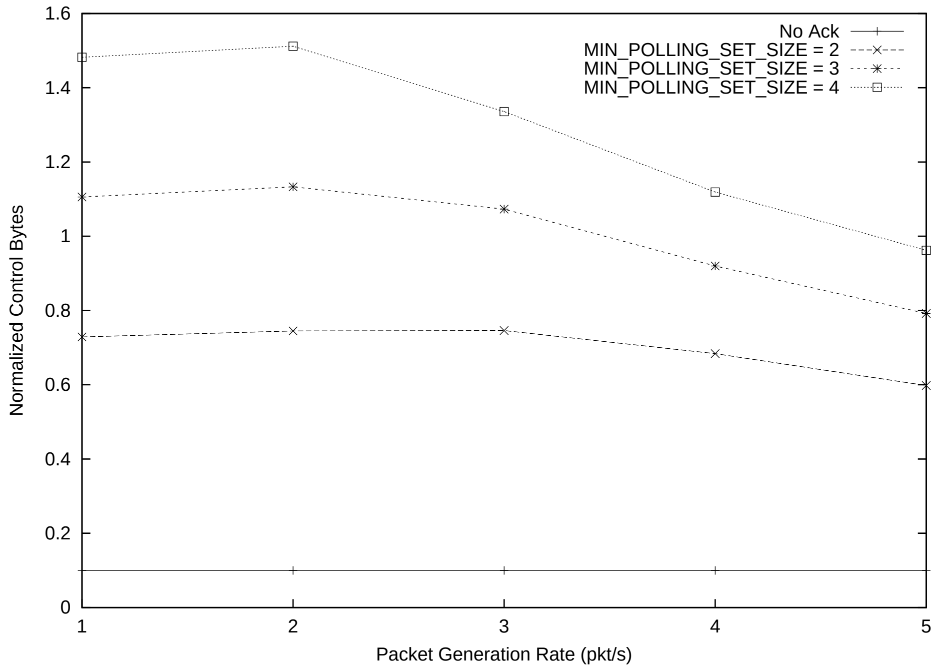

Fig. 34. Normalized control bytes

its delivery fraction drops fast. At the same time, our scheme achieves significantly higher delivery fractions. The figure also illustrates the effects of the minimal polling set size on the performance. When a larger polling set is required, the duration of the polling process has to be longer, which increases the probability that a data frame is corrupted by hidden nodes' transmissions.

In Fig. 33, we show the average end-to-end data packet latency under different traffic loads. Since the network is static, there is no routing delay. We also ignore the CPU processing delay at each intermediate node. Therefore, the end-to-end packet latency here includes queueing delays, retransmission delays and propagation delays. It is shown that, on the average, our scheme has much higher packet latency than unreliable, pure broadcast scheme. This is caused by retransmission and batching. When the minimal polling set size increases, the packet latency increases very fast, especially when traffic load is high. The reason is that a larger polling set means higher probability of transmission failure, which makes each node wait for a longer time before next retry. If user's application has delay constraint, a trade-off on security may be needed.

In Fig. 34, we show the overhead of our scheme under different traffic loads. We use the metric *Normalized control byte overhead*, which is defined as the total bytes of transmitted control data (POLL, REPLY, MAC header) divided by the total bytes of received data payloads by all nodes. For pure broadcast, this overhead is a constant, equal to the size of a MAC header divided by the size of a MAC frame body. It is shown that the normalized control overhead decreases as the traffic load increases. The reason is that, in this case, there tend to be multiple frames in a node's queue, and each polling process can be followed by multiple data transmissions. In other words, each polling is more efficient. Another observation is that the normalized control overhead is high when the minimal polling set size is large. This is because more dummy POLLs may need to be generated to meet the minimal polling set size constraint.

E. Related Work

1. P2P Anonymity Scheme

Crowds [72] is a p2p anonymity scheme that provides source anonymity to web browsing. A user joins a "crowd" by starting a *jondo* process on her computer and contacting with a server, called *blender*. The blender passes the current crowd membership information to the jondo process. This jondo serves as the user's web proxy. Upon receiving a new HTTP request from the user's web browser, the jondo initiates the establishment of a random *path* of jondos that carries the user's transactions to and from the web server. More precisely, the jondo picks a jondo from the crows (possibly itself) at random, and forwards the request to it. When this jondo receives the request, it flips a biased coin to determine whether or not to forward the request to another jondo; the coin indicates to forward with probability p_f. If the result is to forward, then the jondo selects a random jondo and forwards the request to it, and otherwise the jondo submits the request to the end web server for which the request was destined. So, each request travels from the user's browser, through some number of jondos, and finally to the end server. Subsequent requests to the same web server follow the same jondo path, and server replies traverse the same path as the requests, only in reverse. To a web server, it does not know whether the user from whom it receives request is the original requester or a forwarder. Unlike MIX, Crowds does not use layered encryption of request to hide the source from intermediate jondoes. Instead, a request is encrypted using an encryption key shared by all jondoes on the path. This encryption does not protect against an attacker who cooperates with one of the crowd members that are selected to forward the request. Crowds is also vulnerable to trivial traffic analysis attack, since the encrypted requests are forwarded without modification.

Tarzan [27] is a p2p anonymous IP packet relaying scheme. It uses layered encryption, pseudo-random path and dummy traffic to achieve anonymity against a powerful global observer. Tarzan is designed based on a vision of thousands of nodes, distributed on the Internet, volunteering to provide relaying services to each others' traffic. The scheme gives protocols for peer discovery, peer selection, tunnel setup, etc. Unlike Crowds, Tarzan is a self-organizing and fully decentralized system, without a centralized registration and peer discovery server. So it is more scalable than Crowds in serving a large number of users. It is also more general purpose than web-specific Crowds, because it provides anonymity at the IP layer and is transparent to applications.

2. Anonymous Routing Scheme

ANODR [55] is an on-demand anonymous routing protocol for MANET with the objective of developing "untraceable" routes and packet flows. It uses a "route pseudonymity" approach to achieve the goal. An anonymous route discovery process establishes an on-demand route between a source node and a destination node. Each hop en route is associated with a random *route pseudonym.* During the route discovery, Route Request (RREQ) packets and Route Reply (RREP) packets are encrypted and the route information is kept in an *onion* data object, which ensures unlinkability among route pseudonyms. Data forwarding in the network is based on route pseudonyms without revealing the indentities of involved nodes. In each locality, eavesdroppers or any bystander other than the forwarding nodes can only detect the transmission of wireless packets stamped with random route pseudonyms. It is hard for them to trace how many nodes in the locality, who is the transmitter or receiver, where a packet flow comes from and where it goes to (i.e., what are the previous hops and the next hops en route), let alone the source sender and the destination receiver of the flow. The drawback of this protocol is that it cannot utilize many route repair and route optimization techniques in on-demand routing protocols such as DSR and AODV.

Basagni et al. [6] proposed to encrypt routing messages with a network-wide symmetric key. This scheme effectively stops eavesdroppers, but fails when a single node is compromised and discloses the key. The authors argue to protect the key using tamper resistance facilities which introduce tremendous physical cost.

3. Reliable Broadcast Scheme

The batched broadcast technique in the proposed scheme is inspired by some reliable multicast schemes in IEEE 802.11 network, especially the Batch Mode Multicast MAC Protocol (BMMM) [84]. In BMMM, the sender transmits RTS frames before sending the data and transmits Request-to-Acknowledge (RAK) frames after sending the data, to each intended receiver. Receivers transmit CTS frames and ACK frames respectively as replies. The batch of RTS/CTS is similar to the POLL-REPLY sequence in our scheme. The difference is that in BMMM, the RTS frames are transmitted at fixed intervals, while in our scheme, the spacings between consecutive POLL frames are not fixed. The RTS interval in BMMM is larger than $DIFS$ in a IEEE 802.11 DSSS (direct sequence spread spectrum) wireless network. So it does not prevent a neighbor from thwarting a transmission. Our scheme does not have this problem.

CHAPTER V

CONCLUSIONS AND FUTURE WORK

In this dissertation we study how to provide network camouflaging services in wireless networks. We identified two classes of network camouflaging needs: hiding traffic pattern in a network, and hiding communication relationship between network users. The latter is also called anonymity. We discussed essential techniques for satisfying each class of needs. The basic technique for hiding traffic pattern is traffic padding, which introduces dummy traffic into the network. The basic techniques for anonymity include broadcasting, DC-network and Mix-network. While the effectiveness of a technique usually depends on an adversary's ability of launching attacks, the efficiency of a technique is more affected by the nature of networks where the technique is applied. A wireless network has many characteristics that do not exist in wired networks, such as low bandwidth, low power supply, node mobility, etc. Thus, a close examination of existing network camouflaging techniques and evaluating their efficiencies in wireless networks is well justified.

We studied the traffic padding efficiency in a sensor network, where distinct user interests produce different traffic patterns. We assume that each traffic pattern can be linked to a unique user interest. To hide the current user interest, a temporally constant traffic pattern should be maintained and presented to potential observers. We show that this goal can be achieved by two traffic padding schemes, one with source padding, another with link padding. We compare the two approaches in term of the system lifetime and the system energy consumption. Although the linear programming technique can be used to find the optimal values of each metric, it is not a very efficient approach. In this dissertation, we present two Flow Deviation based algorithms to find near-optimal values of each metric. Using the proposed algorithms, we are able to give a quantitative comparison between the two traffic padding schemes. It is shown that link padding can give a longer system lifetime and consumes less energy than source padding. As future work, we are working on new traffic padding model in which the cover traffic pattern is not constant, but adaptive to the actual traffic pattern. An adaptive cover traffic pattern will further reduce the padding cost and improve the system performance, but its implications on security need to be well understood.

We also studied the design of a MIX-network that provides anonymous connection services to mobile users in an ad hoc network. This is a dynamic MIX-net. One of the critical issues is the MIX route algorithm, which has substantial impacts on the network performance. Existing algorithms proposed for static MIX-nets show poor performance in a dynamic MIX-net. The main reason is that they are not adaptive to changes in network topology. To overcome the problem, we propose two adaptive MIX route algorithms. The first algorithm is the Nearest MIX algorithm, which returns the nearest MIX to any mobile node. Using this algorithm, the source node of a connection can forward all its traffic through a MIX in its proximity, instead of a randomly picked, probably distant MIX. We provide two schemes for discovering and maintaining the nearest MIX information. One is on-demand MIX solicitation anycasting scheme. Another is proactive MIX advertisement flooding scheme. The second algorithm is the Destination-initiated MIX Route (DIMR) algorithm. This algorithm can return MIX routes with multiple MIXes, which is more secure than the Nearest MIX algorithm when the network is under active attack. We conducted extensive simulations to evaluate the performance of the proposed algorithms. It is shown that the adaptive algorithms achieve significant performance gain than non-adaptive algorithms in terms of packet delivery fraction and packet latency. For future work, there are several directions. One direction is to integrate MIX route discovery into routing protocol (e.g., DSR [47]) at the network layer to reduce route acquisition delay and routing overhead. Another direction is to improve security of the proposed algorithm. For example, the algorithm can include a mechanism to establish trust among nodes. The objective is that the misbehavior of a node (such as tampering) can be detected through node cooperation and the damage can be reduced to minimal [98].

Finally, we studied the problem of providing anonymity in wireless ad hoc network from a new direction, which has not been explored by other researchers, i.e., link layer solution. It is well known that the IEEE 802.11 MAC protocol provides two data transmission services: reliable unicast and unreliable broadcast. Broadcasting is an effective approach for receiver anonymity. This property can be utilized to achieve the goal of anonymous communication. Specifically, we propose to convert all unicast frames into broadcast frames with the address of the intended receiver being concealed through link layer encryption. If a packet is transmitted anonymously at all links on its route, then it is untraceable by outsider eavesdropper as well as compromised nodes. One shortage of the proposed scheme is unreliability associated with broadcast transmission in 802.11 network. We propose a new MAC protocol to overcome this problem. This protocol uses a polling method to check availability of neighbors, uses a selective repeat ARQ protocol to schedule retransmissions of lost frames, and uses batched transmission to prevent traffic analysis attack. We evaluate the performance of the protocol through simulation. It is shown that, in a static network, our protocol greatly improves the reliability of anonymous transmission. As future work, we plan to evaluate the performance of the protocol in a dynamic network with node mobility.

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: A survey. *Computer Networks (Elsevier)*, 38(4):393–422, Nov. 2002.

[2] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus. Requirements for traffic engineering over MPLS. Internet RFC 2702, http://www.ietf.org/rfc/rfc2702.txt?number=2702, Sept. 1999.

[3] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens. An on-demand secure routing protocol resilient to Byzantine failures. In *Proc. of the ACM Workshop on Wireless Security (WiSe)*, pages 21–30, Atlanta, GA, Sept. 2002.

[4] P. Bahl and V. N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *Proc. of IEEE INFOCOM*, pages 775–784, Tel Aviv, Israel, Apr. 2000.

[5] D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong. Talking to strangers: Authentication in ad hoc wireless networks. In *Proc. of the Network and Distributed Security Symposium (NDSS)*, pages 23–35, San Diego, CA, Feb. 2002.

[6] S. Basagni, K. Herrin, D. Bruschi, and E. Rosti. Secure pebblenets. In *Proc. of the 2nd ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, pages 156–163, Long Beach, CA, Oct. 2001.

[7] U. Berkeley, LBL, USC/ISI, and Xerox-PARC. *The ns Manual (formerly ns Notes and Documentation)*. 2003.

[8] D. Bertsekas and R. Gallager. *Data Networks (Second edition)*. Prentice Hall, 1992.

[9] A. Boukerche, K. El-Khatib, L. Xu, and L. Korba. A novel solution for achieving anonymity in wireless ad hoc networks. In *Proc. of the ACM Workshop on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks (PE-WASUN)*, pages 30–38, Venice, Italy, Oct. 2004.

[10] Hack Canada. The GSM security technical whitepaper for 2002. http://www.hackcanada.com/blackcrawl/cell/gsm/gsm_security.html, Jan. 2002.

[11] P. Castro, P. Chiu, T. Kremenek, and R. Muntz. A probabilistic room location service for wireless networked environments. In *Proc. of the 3rd International Conference on Ubiquitous Computing (Ubicomp)*, pages 18–34, Atlanta, GA, Sept. 2001.

[12] J.-H. Chang and L. Tassiulas. Energy conserving routing in wireless ad-hoc networks. In *Proc. of IEEE INFOCOM*, pages 22–31, Tel Aviv, Israel, Apr. 2000.

[13] Y. J. Chang, J. L. Wu, and H. J. Hu. Optimal virtual circuit routing in computer networks. *IEE Proceedings I: Communications, Speech and Vision*, 139(6):625–632, Dec. 1992.

[14] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, Feb. 1981.

[15] D. Chaum. Security without identification: Transaction systems to make Big Brother obsolete. *Communications of the ACM*, 28(10):1030–1044, 1985.

[16] D. Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of Cryptology*, 1:65–75, 1988.

[17] M. Coates, A. Hero, R. Nowak, and B. Yu. Network tomography: Estimating source-destination traffic intensities from link data. *Journal of the American Statistical Association*, 91(433):365–377, Mar. 1996.

[18] M. Coates, A. Hero, R. Nowak, and B. Yu. Internet tomography. *IEEE Signal Processing Magazine*, 19(3):47–65, 2002.

[19] T. H. Corman, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms.* The MIT Press, 1990.

[20] L. Cottrell. Mixmaster and remailer attack. http://riot.eu.org/anon/doc/remailer-essay.html, 1994.

[21] B. Dahill, B. Levine, E. Royer, and C. Shields. A secure routing protocol for ad hoc networks. Technical Report UM-CS-2001-037, Electrical Engineering and Computer Science, University of Michigan, Aug. 2001.

[22] G. Danezis, R. Dingledine, and N. Mathewson. Mixminion: Design of a type III anonymous remailer protocol. In *Proc. of the IEEE Symposium on Security and Privacy*, pages 28–41, Berkeley, CA, May 2003.

[23] H. Deng, W. Li, and D. P. Agrawal. Routing security in wireless ad hoc network. *IEEE Communications Magazine*, 40(10):70–75, Oct. 2002.

[24] A. Fasbender, D. Kesdogan, and O. Kubitz. Variable and scalable security: Protection of location information in mobile IP. In *Proc. of the 46th IEEE Conference on Vehicular Technology (VTC)*, pages 963–967, Atlanta, GA, Apr. 1996.

[25] H. Federrath, A. Jerichow, D. Kesdogan, A. Pfitzmann, and D. Trossen. Minimizing the average cost of paging on the air interface - an approach considering

privacy. In *Proc. of the 47th IEEE Conference on Vehicular Technology (VTC)*, pages 1253–1257, Phoenix, AZ, May 1997.

[26] L. Fratta, M. Gerla, and L. Kleinrock. The flow deviation method: An approach to store-and-forward communication network design. *Networks*, 3:97–133, 1973.

[27] M. J. Freedman and R. Morris. Tarzan: A peer-to-peer anonymizing network layer. In *Proc. of the 9th ACM Conference on Computer and Communications Security (CCS)*, pages 193–206, Washington, DC, Nov. 2002.

[28] X. Fu, B. Graham, R. Bettati, and W. Zhao. On effectiveness of link padding for statistical traffic analysis attacks. In *Proc. of the 23rd International Conference on Distributed Computing Systems (ICDCS)*, pages 340–347, Providence, RI, May 2003.

[29] X. Fu, Y. Zhu, B. Graham, R. Bettati, and W. Zhao. On flow marking attacks in wireless anonymous communication networks. In *Proc. of the 25th International Conference on Distributed Computing Systems (ICDCS)*, pages 493–503, Columbus, OH, June 2005.

[30] D. Goldschlag, M. Reed, and P. Syverson. Onion routing for anonymous and private internet connections. *Communications of the ACM*, 42(2):39–41, 1999.

[31] M. Gruteser and D. Grunwald. Enhancing location privacy in wireless LAN through disposable interface identifiers: A quantitative analysis. In *Proc. of the 1st ACM Workshop on Wireless Mobile Applications and Services on WLAN Hotspots (WMASH)*, pages 46–55, San Diego, CA, Dec. 2003.

[32] Y. Guan, C. Li, D. Xuan, R. Bettati, and W. Zhao. Preventing traffic analysis for real-time communication networks. In *Proc. of the IEEE Military Commu-*

nication Conference (MILCOM), pages 744–750, Atlantic City, NJ, Oct. 1999.

[33] C. Gulcu and G. Tsudik. Mixing email with Babel. In *Proc. of the IEEE Symposium on Network and Distributed System Security (NDSS)*, pages 2–16, San Diego, CA, Feb. 1996.

[34] W. Heinzelman. *Application-specific Protocol Architectures for Wireless Networks*. Ph.D. dissertation, MIT, 2000.

[35] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Proc. of the 33rd International Conference on System Sciences (HICSS)*, Maui, HI, Jan. 2000.

[36] S. Hoff, K. Jakobs, and D. Kesdogan. Secure location management in UMTS. In *Proc. of the IFIP TC6/TC11 International Conference on Communication and Multimedia Security*, Essen, Germany, Sept. 1996.

[37] T. Hou, Y. Shi, and H. Sherali. On rate allocation in wireless sensor networks with network lifetime requirement. In *Proc. of the 5th ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, Tokyo, Japan, May 2004.

[38] Y.-C. Hu, D. B. Johnson, and A. Perrig. SEAD: Secure efficient distance vector routing in mobile wireless ad hoc networks. In *Proc. of the 4th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*, pages 3–13, Callicoon, NY, June 2002.

[39] Y.-C. Hu, A. Perrig, and D. B. Johnson. Ariadne: A secure on-demand routing protocol for ad hoc networks. In *Proc. of the 8th International Conference on Mobile Computing and Networking (MobiCom)*, pages 12–23, Atlanta, GA,

Sept. 2002.

[40] Y.-C. Hu, A. Perrig, and D. B. Johnson. Zero-interaction authentication. In *Proc. of the 8th International Conference on Mobile Computing and Networking (MobiCom)*, pages 1–11, Atlanta, GA, Sept. 2002.

[41] Y.-C. Hu, A. Perrig, and D. B. Johnson. Rushing attacks and defense in wireless ad hoc network routing protocols. In *Proc. of the ACM Workshop on Wireless Security (WiSe)*, pages 30–40, San Diego, CA, Sept. 2003.

[42] J.-P. Hubaux, L. Buttyan, and S. Capkun. The quest for security in mobile ad hoc networks. In *Proc. of the 2nd ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, Long Beach, CA, Oct. 2001.

[43] E. Hughes. A cypherpunk's manifesto. http://www.activism.net/cypherpunk, Mar. 1993.

[44] IEEE. *IEEE std 802.11, 1999 Edition, Wireless LAN Medium Access Control (MAC) and Phyiscal Layer (PHY) Specifications.* 1999.

[45] ISO. *Information Process Systems - Open System Interconnection - Basic Reference Model - ISO 7498.* American National Standards Association, Inc., New York, NY, 1984.

[46] ISO. *Information Process Systems - Open System Interconnection - Proposed Draft Addendum 2 - ISO 7498.* American National Standards Association, Inc., New York, NY, 1988.

[47] D. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. In T. Imielinski and H. Korth, editors, *Mobile Computing*, volume 353, pages 153–181. Kluwer Academic Publishers, 1996.

[48] S. Kent and R. Atkinson. IP encapsulating security payload (ESP). Internet RFC 2406, http://www.ietf.org/rfc/rfc2406.txt?number=2406, Nov. 1998.

[49] S. Kent and R. Atkinson. Security architecture for the Internet protocol. Internet RFC 2401, http://www.ietf.org/rfc/rfc2401.txt?number=2401, Nov. 1998.

[50] D. Kesdogan, J. Egner, and R. Büschkes. Stop-and-Go MIXes: Providing probabilistic anonymity in an open system. In *Proc. of the Information Hiding Workshop (IH)*, pages 83–98, Portland, OR, Apr. 1998.

[51] D. Kesdogan, H. Federrath, A. Jerichow, and A. Pfitzmann. Location management strategies increasing privacy in mobile communication systems. In *Proc. of the 12nd International Information Security Conference (IFIP/SEC)*, pages 39–48, Samos, Greece, May 1996.

[52] D. Kesdogan and X. Fouletier. Secure location information management in cellular radio systems. In *Proc. of the IEEE Wireless Communications Systems Symposium (WCSS)*, pages 35–40, New York, NY, Nov. 1995.

[53] H. Kim, Y. Seok, N. Choi, Y. Choi, and T. Kwon. Optimal multi-sink positioning and energy-efficient routing in wireless sensor networks. In *Proc. of the International Conference on Information Networking (ICOIN)*, pages 264–274, Jeju Island, Korea, Jan. 2005.

[54] J. Kong. *Anonymous and Untraceable Communications in Mobile Wireless Networks.* Ph.D. dissertation, University of California at Los Angeles, 2004.

[55] J. Kong and X. Hong. ANODR: Anonymous on demand routing with untraceable routes for mobile ad-hoc networks. In *Proc. of the 4th ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, pages 291–302, An-

napolis, MD, June 2003.

[56] A. M. Ladd, K. E. Bekris, A. Rudys, L. E. Kavraki, D. S. Wallach, and G. Marceau. Robotics-based location sensing using wireless Ethernet. In *Proc. of the 8th International Conference on Mobile Computing and Networking (MobiCom)*, pages 227–238, Atlanta, GA, Sept. 2002.

[57] S. Marti, T. J. Giuli, K. Lai, and M. Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Proc. of the 6th International Conference on Mobile Computing and Networking (MobiCom)*, pages 255–265, Boston, MA, Aug. 2000.

[58] D. Martin. *Local Anonymity in the Internet.* Ph.D. dissertation, Boston University, 1999.

[59] R. E. Newman-Wolfe and B. R. Venkatraman. High level prevention of traffic analysis. In *Proc. of the 7th Annual Computer Security and Applications Conference*, pages 102–109, San Antonio, TX, Dec. 1991.

[60] P. Papadimitratos and Z. J. Hass. Secure routing for mobile ad hoc networks. In *Proc. of SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS)*, pages 193–204, San Antonio, TX, 2002.

[61] P. Papadimitratos and Z. J. Hass. Secure link state routing for mobile ad hoc networks. In *Proc. of the IEEE Workshop on Security and Assurance in Ad Hoc Networks (WSAAN)*, pages 27–31, Orlando, FL, Jan. 2003.

[62] V. D. Park and M. S. Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. In *Proc. of IEEE INFOCOM*, pages 1405–1413, Kobe, Japan, Apr. 1997.

[63] C. E. Perkins. Ad-hoc on-demand distance vector routing. In *IEEE Military Communication Conference (MILCOM) panel on Ad Hoc Networks*, Monterey, CA, Nov. 1997.

[64] C. E. Perkins and E. M. Royer. Ad-hoc on-demand distance vector routing. In *Proc. of the 2nd IEEE Workshop on Mobile Computer Systems and Applications (WMCSA)*, pages 90–100, New Orleans, LA, Feb. 1999.

[65] A. Pfitzmann and M. Köhntopp. Anonymity, unobservability, and pseudonymity: A proposal for terminology. In H. Federrath, editor, *Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, LNCS 2009, pages 1–9, Heidelberg, Germany, July 2000. Springer-Verlag.

[66] A. Pfitzmann, B. Pfitzmann, and M. Waidner. ISDN-Mixes: Untraceable communication with very small bandwidth overhead. In *Proc. of the 7th International Information Security Conference (IFIP/SEC)*, pages 451–463, Mannheim, Germany, Feb. 1991.

[67] A. Pfitzmann and M. Waidner. Networks without user observability – design options. In *Proc. of EUROCRYPT*, pages 245–253, Linz, Austria, Apr. 1985.

[68] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The Cricket location-support system. *Mobile Computing and Networking*, 10(1):32–43, 2000.

[69] B. Radosavljevic and B. Hajek. Hiding traffic flow in communication networks. In *Proc. of the IEEE Military Communication Conference (MILCOM)*, pages 1096–1100, San Diego, CA, Oct. 1992.

[70] T. S. Rappaport. *Wireless Communications: Principles and Practice (second*

edition). Prentice Hall, 1996.

[71] J.-F. Raymond. Traffic analysis: Protocols, attacks, design issues and open problems. In H. Federrath, editor, *Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, LNCS 2009, pages 10–29, Heidelberg, Germany, July 2000. Springer-Verlag.

[72] M. K. Reiter and A. D. Rubin. Crowds: Anonymity for Web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, June 1998.

[73] M. Rennhard and B. Plattner. Introducing MorphMix: Peer-to-peer based anonymous internet usage with collusion detection. In *Proc. of the Workshop on Privacy in the Electronic Society (WPES)*, pages 91–102, Washington, DC, Nov. 2002.

[74] E. M. Royer and C.-K. Toh. A review of current routing protocols for ad-hoc mobile wireless networks. *IEEE Personal Communications Magazine*, 6(2):46–55, Apr. 1999.

[75] K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer. A secure routing protocol for ad hoc networks. In *Proc. of the 10th IEEE International Conference on Network Protocols (ICNP)*, pages 78–87, Paris, France, Nov. 2002.

[76] A. Serjantov, R. Dingledine, and P. Syverson. From a trickle to a flood: Active attacks on several MIX types. In F. Petitcolas, editor, *Proc. of the Information Hiding Workshop (IH)*, LNCS 2578, Noordwijkerhout, The Netherlands, Oct. 2002. Springer-Verlag.

[77] A. Serjantov and P. Sewell. Passive attack analysis for connection-based

anonymity systems. In *Proc. of the 8th European Symposium on Research in Computer Security (ESORICS)*, pages 116–131, Gjovik, Norway, Oct. 2003.

[78] R. Sherwood, B. Bhattacharjee, and A. Srinivasan. P5: A protocol for scalable anonymous communication. In *Proc. of the IEEE Symposium on Security and Privacy (SP)*, pages 58–70, Oakland, CA, May 2002.

[79] S.-T. Sheu, Y. Tsai, and J. Chen. A highly reliable broadcast scheme for IEEE 802.11 multi-hop ad hoc networks. In *Proc. of the IEEE International Conference on Communications (ICC)*, pages 610–615, New York, NY, Apr. 2002.

[80] W. Si and C. Li. RMAC: A reliable multicast MAC protocol for wireless ad hoc networks. In *Proc. of the 33rd International Conference on Parallel Processing (ICPP)*, pages 494–501, Montreal, Canada, Aug. 2004.

[81] A. Smailagic and D. Kogan. Location sensing and privacy in a context-aware computing environment. *IEEE Wireless Communications*, 9(5):10–17, Oct. 2002.

[82] F. Stajano and R. Anderson. The resurrecting duckling: Security issues for ad hoc wireless networks. In *Proc. of the 7th International Workshop on Security Protocols*, LNCS 1796, pages 172–194, Cambridge, UK, Apr. 1999. Springer-Verlag.

[83] W. Stallings. *Cryptography and Network Security: Principles and Practice (second edition)*. Prentice Hall, 1998.

[84] M. T. Sun, L. Huang, A. Arora, and T. H. Lai. Reliable MAC layer multicast in IEEE 802.11 wireless networks. In *Proc. of the 31st International Conference*

on Parallel Processing (ICPP), pages 527–536, Vancouver, Canada, Aug. 2002.

[85] P. F. Syverson, D. M. Goldschlag, and M. G. Reed. Anonymous connections and onion routing. In *Proc. of the IEEE Symposium on Security and Privacy (SP)*, pages 44–54, Oakland, CA, Dec. 1997.

[86] A. S. Tanenbaum. *Computer Networks (Third edition).* Prentice Hall, 1996.

[87] K. Tang and M. Gerla. MAC layer broadcast support in 802.11 wireless networks. In *Proc. of the IEEE Military Communication Conference (MILCOM)*, pages 544–548, Los Angeles, CA, Oct. 2000.

[88] K. Tang and M. Gerla. MAC reliable broadcast in ad hoc networks. In *Proc. of the IEEE Military Communication Conference (MILCOM)*, pages 1008–1013, McLean, VA, Oct. 2001.

[89] B. Timmerman. A security model for dynamic adaptive traffic masking. In *Proc. of the 1997 Workshop on New Security Paradigms*, pages 1–25, Langdale, Cumbria UK, Sept. 1997.

[90] B. Timmerman. Secure dynamic adaptive traffic masking. In *Proc. of the 1999 Workshop on New Security Paradigms*, pages 13–24, Ontario, Canada, Sept. 1999.

[91] J. Tourrihes. Robust broadcast: Improving the reliability of broadcast transmissions on CSMA/CA. In *Proc. of the 9th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Boston, MA, Sept. 1998.

[92] B. R. Venkatraman. *Prevention of Traffic Analysis and Associated Covert Channels.* Ph.D. dissertation, University of Florida, 1994.

[93] V. L. Voydock and S. T. Kent. Security mechanisms in high-level network protocols. *ACM Computing Surveys*, 15(2):135–171, June 1983.

[94] R. Want, A. Hopper, V. Falcao, and A. Gibbons. The active badge location system. *ACM Transactions on Information Systems*, 10(1):91–102, 1992.

[95] Wikipedia. Penet remailer. http://en.wikipedia.org/wiki/penet_remailer, Aug. 2005.

[96] X. Wu and B. Bhargava. AO2P : Ad hoc on-demand position-based private routing protocol. *IEEE Transactions on Mobile Computing*, 4(4):335–348, July 2005.

[97] K. Xu, M. Gerla, and S. Bae. How effective is the IEEE 802.11 RTS/CTS handshake in ad hoc networks? In *Proc. of the IEEE Global Telecommunication Conference (GLOBECOM)*, pages 17–21, Taipei, Taiwan, Nov. 2002.

[98] H. Yang, X. Meng, and S. Lu. Self-organized network layer security in mobile ad hoc networks. In *Proc. of the ACM Workshop on Wireless Security (WiSe)*, pages 11–20, Atlanta, GA, Sept. 2002.

[99] S. Yi and R. Kravets. Key management for heterogeneous ad hoc wireless networks. Technical Report UIUCDCS-R-2002-2290, Dept. of Computer Science, Univ. of Illinois at Urbana-Champaign, July 2002.

[100] M. G. Zapata and N. Asokan. Securing ad hoc routing protocols. In *Proc. of the ACM Workshop on Wireless Security (WiSe)*, pages 1–10, Atlanta, GA, Sept. 2002.

[101] L. Zhou and Z. J. Haas. Securing ad hoc networks. *IEEE Networks*, 13(6):24–30, Nov. 1999.

[102] S. Zhu, S. Xu, S. Setia, and S. Jajodia. LHAP: A lightweight hop-by-hop authentication protocol for ad hoc networks. In *Proc. of the 23rd International Conference on Distributed Computing Systems Workshop (ICDCSW)*, pages 749–757, Providence, RI, May 2003.

[103] Y. Zhu, X. Fu, B. Graham, R. Bettati, and W. Zhao. On flow correlation attacks and countermeasures in mix networks. In *Proc. of the Workshop on Privacy Enhancing Technologies (PET)*, pages 207–225, Toronto, Canada, May 2004.

VITA

Shu Jiang received his B.E. degree in computer science from University of Science and Technology, China, in 1990 and his M.E. degree in computer science from Nanjing University, China, in 1993. From 1993 to 1995, he worked for Lenovo Group, Ltd., as an engineer, at Beijing, China. From 1998 to 2003, he worked for Texas Engineering Experiment Station, a state agency of Texas, as a Systems Analyst and Development Coordinator. He enrolled into the Ph.D. program at the Department of Computer Science, Texas A&M University, in 1998. His research areas cover security issues in sensor networks and wireless mobile ad hoc networks. He is a student member of IEEE and ACM.

The typist for this dissertation was Shu Jiang.