uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Alireza Ali Akbar Nezhad
AUTEUR DE LA THÈSE / AUTHOR OF THESIS

Ph.D. (Electrical Engineering)
GRADE / DEGREE

School of Information Technology and Engineering
FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Anonymous and Location-concealing Routing for Multihop Ad hoc and Sensor Networks
TITRE DE LA THÈSE / TITLE OF THESIS

Dimitrios Makrakis
DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

Ali Miri
CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Abderrahim Benslimane
(University d'Avignon)

Ashraf Matrawy

Shervin Shirmohammadi

Jiying Zhao

Gary W. Slater
Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

Anonymous and Location-Concealing Routing for Multihop Ad hoc and Sensor Networks

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the PhD degree in Electrical and Computer Engineering

School of Information Technology and Engineering
Faculty of Engineering and Computer Science
University of Ottawa

Your file Votre référence
ISBN: 978-0-494-61383-2
Our file Notre référence
ISBN: 978-0-494-61383-2

ABSTRACT

Anonymity and location privacy are two important aspects of network security for users. This thesis is concerned with location privacy and anonymity in multihop wireless ad hoc networks. More specifically, it proposes techniques to protect these features from being violated as a by-product of ad hoc routing protocols. We have proposed a novel destination-centric routing approach called Destination Controlled Routing (DCR) in which the destination is the master and controls how packets are delivered to it so that it remains unknown. In addition, DCR is designed in a way that it also preserves the location privacy and anonymity of the source node as well as the communication anonymity. Unlike other anonymous routing protocols for ad hoc networks, DCR is based on the proactive as opposed to the reactive routing approach.

We have instantiated DCR for various kinds of multihop ad hoc networks. For networks of type mesh and MANET, we have proposed three variations of our V-routing protocol based on this approach. We have also adapted V-routing to multicasting applications. Given a sufficient number of data flows in the network, V-routing can protect location privacy and communication anonymity against a global eavesdropper. It achieves this goal by using a transient destination node and hop-by-hop anonymous routing between that node and the real destination node in a way that the location of the real destination node does not need to be revealed.

For multihop wireless sensor networks, we have developed different versions of our DCARPS (Destination Controlled Anonymous Routing Protocol for Sensornets) protocol for high and low volume traffic load situations. Unlike most routing protocols

for sensor networks, our DCARPS protocols do not use the sink-originated beacons that reveal the location of the sink. Finally, in order to accommodate the proactive nature of DCR, we have proposed a new low-overhead topology discovery protocol called 2hop-clustered NTDP based on link state routing.

We have also provided a formal security analysis for evaluating the security performance of any randomized routing protocol using an information-theoretical method based on entropy. We have applied this method to DCARPS and showed that it has a good performance in terms of location privacy.

To my wife

Sahar Kajbaf

for standing by me throughout these years

TABLE OF CONTENTS

LIST OF ACRONYMS

ACK	Acknowledgment
AD	Access router of Destination node
ANODR	ANonymous On Demand Routing
AnonDSR	Anonymous Dynamic Source Routing
AODV	Ad hoc On demand Distance Vector
AO2P	Ad hoc On demand Position-based Private
AS	Access router of Source node
CA	Certificate Authority
CoA	Care of Address
CPU	Central Processing Unit
CTSR	Cost To Success Ratio
DCARPS	Destination Controlled Anonymous Routing Protocol for Sensornets
DCR	Destination Controlled Routing
DHCP	Dynamic Host Configuration Protocol
DI	Downstream Incoming
DoS	Denial of Service
DSDV	Destination-Sequenced Distance Vector
DSR	Dynamic Source Routing
ECG	Electrical CardioGram
FCC	Federal Communications Commision
FSR	Fisheye State Routing
GPS	Global Positioning System

HLR	Home Location Register
HSR	Hierarchical State Routing
IETF	Internet Engineering Task Force
IP	Internet Protocol
KDC	Key Distribution Center
LDAP	Light-weight Directory Access Protocol
LSA	Link State Advertisement
MAC	Medium Access Control
MANET	Mobile Ad hoc NETwork
MIB	Multicast Information Base
MIP	Mobile Internet Protocol
MPR	Multi-Point Relay
MPLS	Multi-Protocol Label Switching
MWIS	Maximal Weighted Independent Set
NAT	Network Address Translation
NDM	Non-Disclosure Method
NTDP	Network Topology Discovery Protocol
OLSR	Optimized Link State Routing
OSI	Open Systems Interconnection
OSPF	Open Shortest Path First
PKI	Public Key Infrastructure
PPR	Privacy Preserving Routing
QoS	Quality of service

RDIS	Route DIScovery
RERR	Route ERRor
RREP	Route REPly
RREQ	Route REQuest
RFC	Request For Comment
SAR	Security aware Ad hoc Routing
RFID	Radio Frequency IDentifier
RIP	Routing Information Protocol
RP	Rendezvous Point
SDAR	Secure Distributed Anonymous Routing
STAR	Source Tree Adaptive Routing
TBRPF	Topology Broadcast based on Reverse Path Forwarding
TCP	Transmission Control Protocol
TTL	Time To Live
UDP	User Datagram Protocl
URL	Uniform Resource Locator
UUID	Universally Unique IDentifier
VHR	Virtual Hop Routing
VLR	Visitor Location Register
WLAN	Wireless Local Area Network
WISP	Wireless Internet Service Provider
WSN	Wireless Sensor Network
ZRP	Zone Routing Protocol

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ACKNOWLEDGMENTS

I am eternally indebted to my supervisors Dr. Ali Miri and Dr. Dimitrios Makrakis for their continuous guidance and support. I am greatly appreciative of the amount of time and energy they dedicated to help me with my research, publications, thesis write-up and every other aspect of my work. Their assistance and direction has been paramount in bringing this dissertation to this point.

My gratitude goes to the members of my dissertation committee Dr. Ioannis Lambadaris, Dr. Jiying Zhao, Dr. Abderrahim Benslimane, Dr. Shervin Shirmohammadi and Dr. Ashraf Matrawy for their constructive feedback. I used their comments to improve my thesis.

I would like to thank a few people at Bell University Labs for their financial and technical support, in particular Mike Milton, Noel Tin and Vanessa Vogwil.

And last but not least, I would like to offer countless thanks to my lovely wife for her understanding, sacrifices, amazing patience and support. I am forever in her debt.

CHAPTER I

INTRODUCTION AND OVERVIEW

In this chapter, we offer an overview of the research area of interest. In particular, we provide a statement of the problem that we have tackled throughout this thesis. We also provide a brief sketch of our solution for the stated problem. We mention the contributions that we have made to this field of research. And finally, we give the outline of the thesis.

Ad hoc networks are a special kind of wireless networks that can be spontaneously formed and used by wireless devices without the need for any infrastructure such as base stations or access points. This kind of network has been in use by militaries for a long time especially in tactical missions where foot soldiers, officers, tanks, planes and so on can communicate with each other in locations where no infrastructure is available. Ad hoc networks are also very useful in emergency operations, law enforcement and rescue missions. In the recent years, these networks have also found many civilian and commercial applications because of their relatively low cost, ease of use and ease of deployment and configuration, for example in meeting rooms, conventions, sensor networks, etc.

Two types of ad hoc networks exist: *single-hop* and *multihop*. In single-hop ad hoc networks, connections are normally established directly between a sender and a receiver node. In some single-hop networks such as Bluetooth, packets may be sent from

the sender to a middle node sometimes called the master or the base station who then relays that packet to the receiver, in which case the nodes form a star with the master at the center. Single-hop ad hoc networks are usually used to cover a small physical area such as a meeting or a convention where almost all nodes are within the coverage range of each other.

Using multihop ad hoc routing protocols, a mobile node can connect to another node even if it is not within its coverage area. This is achieved through cooperation of intermediary nodes that relay packets towards the destination. Such a connection may encompass multiple links (*hops*). Within a multihop ad hoc network, a major challenge is routing packets from the source to the destination as will be explained later. Routing is not needed in single-hop networks as each sending node is either connected directly to the receiving node or to the base station. In Multihop wireless ad hoc networks, routing protocols fall into either *proactive* or *reactive* categories. Proactive routing protocols, also known as table-driven, constantly keep an up-to-date routing table at each node, containing enough information to enable the source node to send its packets to the destination. On the other hand, reactive routing protocols, also known as on-demand, obtain this information only when a route from the source to the destination is needed.

Another important aspect of ad hoc networks is security. The fast and widely spread growth in the use of wireless devices has brought new security concerns to the forefront of wireless communications technology. Two of the most important aspects of security from the users' perspective are *anonymity* and *privacy*. While being very important, these are illusive, controversial and multifaceted concepts that find different meanings in different contexts. For example, in cellular systems, location privacy is

mostly about preventing unauthorized access to the databases of HLR (Home Location Register) and VLR (Visitor Location Register) as they contain information about the current location of the mobile phone while in Chaum's Mixes [1], anonymity and privacy mean hiding the originating address of an electronic mail.

Security is sometimes categorized as either *content* security or *contextual* security. In the context of communications, content security means protecting transmitted data against eavesdropping and manipulation and is usually achieved using encryption and message digests. Contextual security on the other hand is concerned with leakage of information about other aspects of communication such as identities of the parties involved, their locations, frequency and the times of communications, etc. Such invasions can seriously violate the privacy rights of users. Examples of such intrusions are well documented and some of them can be found in [2]. These concerns are usually addressed by *anonymous communication* protocols.

Loosely defined, anonymity in a connection means that a particular message cannot be traced back to its originator or its recipient. However, there is little consensus on what it means to be anonymous. Some widely accepted definitions of anonymity were first given by Pfitzmann and Waidner [3]. They define three types of anonymity; *sender* anonymity, *receiver* anonymity and *unlinkability* anonymity[1].

Definition I.1: Sender anonymity means that the identity of the party who sends a message is hidden.

Definition I.2: Receiver anonymity means that the identity of the receiver of a message is hidden.

[1] Throughout this thesis, we use the terms sender, originator, source and initiator interchangeably. Similarly, we use the terms receiver, recipient and destination interchangeably.

Definition I.3: Unlinkability anonymity means that the sender and the receiver of a message may be identified as being involved in *some* communication but cannot be identified as communicating with each other.

These are important concerns from the point of view of users. For example, the privacy of a user may be violated if it is revealed that he/she is using certain services. Unfortunately, due to the lack of a universal agreement on the meaning of anonymity and privacy, over the past several years, many anonymous systems have been proposed that put forward different and sometimes even conflicting concepts of anonymity and privacy. The shear number of anonymity schemes along with almost absolute absence of formal specifications for them makes understanding all of them an overwhelming challenge. Recently, a first step in formalizing the information hiding properties of such protocols was taken in [4]. This paper defines anonymity as secrecy of identity and privacy as secrecy of relationship. For example, hiding the name of a person in his/her daily activities ensures his anonymity while hiding the fact that he/she is a patient of a certain medical clinic ensures his/her privacy. It also shows that anonymity and privacy are independent i.e. revealing the identity of an item alone does not violate its privacy. In other words, anonymity is neither necessary nor sufficient for privacy.

In this thesis, we are concerned with ***location privacy and user anonymity in the context of routing protocols for multihop wireless ad hoc networks***. There are many ways through which the identity and/or the location of the user of a mobile device may be revealed including physical layer means (e.g. RF fingerprinting), data transmitted on the link layer, traffic analysis, unauthorized access to the databases of context-aware

applications containing location samples, location-dependent temporary IP addresses, and user identification or locating at the time of association with the network, to name a few.

In many cases, this is even desirable, for example in location-aware applications such as Radio Frequency ID (RFID) systems, Active Badge Systems [5] and PARCTAB [6]. As another example, the FCC E911 Phase II mandate requires wireless carriers to identify the location of mobile units making emergency calls within a radius of no more than 125 meters [7]. These applications usually use signal strengths received by fixed access points or base stations in the case of cellular networks as well as other measurements at the air interface layer to estimate the current location of a mobile device. However, this could sometimes be a security concern. Context-aware applications face the challenge of controlling access to user location information gathered from a wireless network and stored in databases. They involve policies and technologies that limit access to this kind of sensitive information by unauthorized parties. For example, the authors of [8] propose the use of a data perturbation algorithm to reduce the accuracy of location samples gathered in a vehicular system before the location server releases that information to a third party application even after the data has been anonymized. The mix zones method [9] provides location privacy for location-based applications monitoring slower users.

We are concerned with preventing the locations and the identities of the communicating users from becoming known to others as a by-product of inherent functions of routing protocols in multihop ad hoc networks. In our terminology, the first attribute is referred to as *location privacy* and the second attribute as *communication anonymity* as defined below:

Definition I.4: Location privacy is defined as the unlinkability between the location and the identity of a user.

Definition I.5: Communication anonymity is defined as the inability of a third party in finding out the identities of both ends of a certain connection.

A large number of routing protocols have been designed for wireless ad hoc networks but as we will explain later all of them are inadequate in terms of security features especially location privacy and anonymity. The bulk of research in this area has been dedicated to making these protocols faster, more efficient, more reliable and more scalable. Little attention has been paid to the security needs of these protocols. In this thesis, we intend to remedy this situation.

I.1 CONTEXUAL SECURITY IN AD HOC NETWORKS

Lack of location privacy (in other words movement traceability) is a big concern in mobile networks and can have serious consequences for users. Location privacy protection act of the United States [10] identifies risks associated with disclosure of location information. Although this act pertains to location-based services and applications, the risks mentioned in the document are relevant in other areas where location privacy is of interest too. The following are some specific examples:

- People going to specific doctors' offices may reveal their ailments.
- People attending special meetings such as AA[2] may reveal their personal problems.
- People's religious and political affiliations may be known by the places to which they go.

[2] Alcoholic Anonymous

- This information may be a basis for court subpoenas.
- Users may become targets of unsolicited advertisement.
- Network topology disclosure is a big concern. The location information obtained for each node can be used to generate a map of the network. It would be a disaster if for example criminals could find out the locations of all police cars and police officers that form an ad hoc network.
- In military situations, which are still the main area of application for ad hoc networks, the locations of personnel and equipment (e.g. robot soldiers, sensors, cameras, etc.) must be kept hidden from the enemy. As another military example, the two opposing sides may be communicating and negotiating a ceasefire but they may not want their locations to be known.

Communication anonymity concerns in ad hoc networks are similar to those of the traditional networks. Consider that an enemy can find out that *generals* of the opposing army are communicating more than usual. Even without deciphering the conversations (if they are encrypted), the enemy can infer that an attack may be imminent. As another example, the fact that two *CEO's* are talking to each other may be an indication that a merger between their respective companies may be being planned.

Location privacy and anonymity are of particular interest in health care. Remote patient monitoring and home care are becoming very important services. In remote monitoring applications, a sensor device is attached to the patient and can be easily used to track his/her movements either as the source or as the destination of a connection. Anonymity of the communication between the patient and the health service provider

must be preserved so that a third party cannot obtain sensitive personal information about the patient. In the home care application, a health care professional equipped with a mobile device may pay a visit to the patient's residence. Tracking his movements could reveal the identities and locations of his patients. These were just a few practical examples to highlight the importance of location privacy and communication anonymity as security services in ad hoc networks.

I.2 PROBLEM STATEMENT

The early ad hoc routing protocols [11-16, 17-31] were designed without security in mind. One of the aspects of security that has been neglected in these routing protocols is *user's location privacy*. Mobile users are not stationary and tracking of their movements through monitoring of their communications is a real concern. In proactive routing protocols all nodes learn about the location of every other node in the network, e.g. the shortest path to reach it, through routing updates. In some cases where Global Positioning System (GPS) is used such as geographical routing protocols, even the geographical coordinates of the node are known. Even in DSDV [11] where each router only keeps a next hop and the smallest hop-count for each destination it is easy to find the location of any destination. A mobile node can constantly listen to local broadcasts, moving from one router to the next, always following the route of the smallest hop-count until it reaches the destination.

In reactive routing protocols the location of the source and the destination are revealed in a different way usually due to route discovery messages. In DSR [17, 18] for example, the full path between the source and the destination is carried in the route

request packets, route reply packets and data packets. Therefore, the source learns about the exact path to the destination and vice versa; moreover all the nodes on the path will have that information about both the source and the destination. In AODV [20, 21], the full path information is not contained in the packets but exists in the routing tables of the intermediate nodes. One of the ways an adversary can learn the locations of the source and the destination in AODV is to listen to route reply messages. These messages are unicasted from the destination back to the source and identify the source address and the destination address as well as the per-hop sender and receiver addresses. By tracing these messages hop-by-hop towards the source/destination an adversary can arrive at the source/destination. Hop-by-hop packet tracing is possible by following a known packet (or a packet stream) after each re-transmission (or new packet arrival). The immediate sender of a packet at each hop can be identified using signal localization techniques such as tri-angulation, tri-lateration and angle-of-arrival. The attacker does not even need to capture or hack into any nodes. AODV as well as TORA [23] rely also on locally broadcasted Hello messages, which reveal the location of a node.

Another security-related shortcoming of regular ad hoc routing protocols is their lack of *communication anonymity*. Normally, some sort of a network ID (*identifier*, such as MAC address or IP address) is used to uniquely identify a device in the network. The network identifiers of the source and the destination nodes are carried in the data packets for routing purposes and also in the route discovery and route reply messages in the case of reactive protocols. Therefore, the anonymity of the communication is violated as well. Moreover, in on-demand routing protocols communication anonymity is violated because

all nodes in the network learn of the fact that the source node is trying to find a path to the destination.

Movement tracking by taking advantage of shortcomings of routing protocols is much easier, less costly, more covert and safer than using other means. It does not require the attacker to own and set up many monitoring points and the attacker does not have to necessarily be present in the transmission range of the target node. For example, a wireless sensor (e.g. a Bluetooth device in undiscoverable mode) can be planted on someone or something and be tracked using RREQ messages by the monitoring party who is the only one aware of the identity of the sensor. Many more possible attacks have been described in the literature against each specific ad hoc routing protocol.

One solution for destination-node privacy may be for the receiver device not to respond to connection requests received from untrusted devices. This means for example that the destination must ignore route discovery messages in on-demand routing protocols (we will also have to disallow route replies from intermediate nodes, which is difficult to control.). But, this solution means that the user would not be able to enjoy for instance many of useful peer-to-peer applications (file sharing, remote monitoring, etc.) and other potentially useful anonymous connections. In effect, this means that the node would be disconnected from the rest of the world most of the time.

Moreover, even if there is a level of trust between the end devices, the destination may still want to hide its current location; for example to avoid compromised nodes in military scenarios or stolen devices that are used as the source or simply because it does not want to make all its movements known even to a trusted party. Imagine if a person could always find out what places his/her partner visits! A security association would

probably exist between these devices yet the destination node may desire to keep its location a secret. As another example, a thief could keep track of the location of a wireless user in an attempt to know when s/he is away from her/his house. On a smaller geographical scale, within an ad hoc network in a shopping mall or a university campus, movements of a user and places or people that s/he visits can be monitored. Intelligent software (especially in the hands of those who have more authority or can gather other types of information that can be cross-referenced with the location information) can infer valuable information from this data. As yet another example, consider a private investigator that monitors the movements of two target suspects equipped with wireless devices. He can find out when and where they meet just by tracking movements of both of them.

Location privacy and communication anonymity are obviously required for ad hoc networks in military scenarios. However, there are many non-military situations where at least one of these properties is needed as well. For example, consider the following two scenarios that demonstrate the need for both of these services at the same time.

Example Scenario 1

A team of employees in a company has been assigned to work on a top-secret project. No one, not even other company employees should know about this project. In addition to employee workstations, one or more servers have been dedicated to this task. To reduce costs and increase efficiency, the company is using wireless networks throughout its large premises. Users are equipped with wireless devices and many access points are installed across the company. While working on this project, team members constantly

collaborate electronically as well as connect to the servers. Obviously, the locations and the identities of the servers must remain hidden from every one; perhaps even from team members. Besides, the locations and the identities of user devices (both wireless and fixed) must be kept a secret so that no one (*adversary*) can physically identify members of the team. Moreover, for personal reasons, no employee would like other employees to be able to track their movements. This scenario is depicted in Figure I-1.

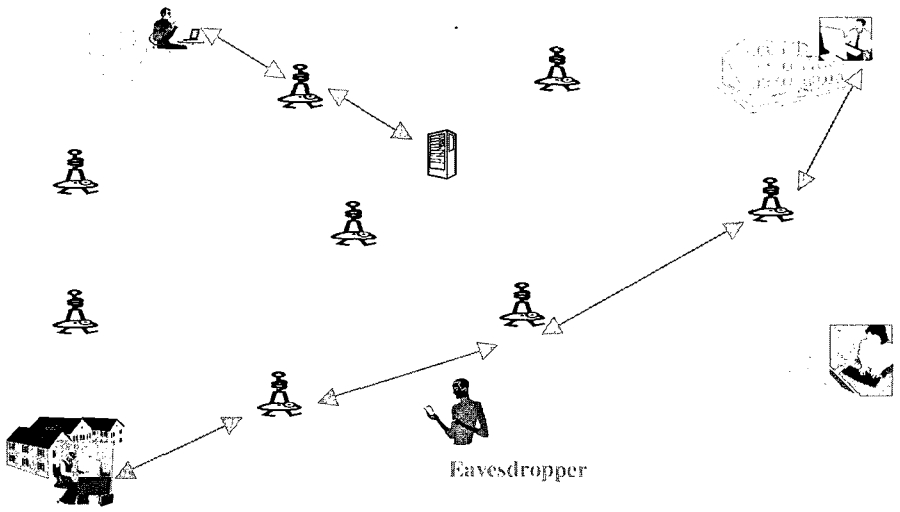


Figure I-1 Importance of location privacy and anonymity in an enterprise network

Imagine that a malicious eavesdropper happens to capture a packet, which he somehow knows belongs to the project. If he can trace the packet to the sender or the receiver he can physically identify members of the secret team or the servers. An anonymous routing protocol must protect against this possibility. On the other hand, imagine that an attacker has somehow identified an entity (a server or a team member) in the project. Now, if the routing protocol does not support communication anonymity, the

attacker will be able to identify other entities potentially related to the project just by listening to the communications of that known entity. Without adequate location privacy support, the adversary may be able to locate the other entities by posing as the identified entity as well

If only the logical ID of a team member is known and the routing protocol does not support location privacy, an attacker (using a compromised node) can find the real identity of that person using the normal operations of the routing protocol to physically locate him. Therefore, privacy with regards to other users (*peer privacy*) is necessary. If wiretapping of the access points (routers) by adversaries is a concern, then the routing protocol must support privacy with regards to the network (*network privacy*) too, i.e. the routers must not be able to identify the senders and receivers either.

Example Scenario 2

Another area where anonymous routing becomes very important is convergence of mobile voice and data networks. With availability of WiFi capable cell phones, VoFi (Voice over WiFi) is poised to revolutionize cellular telephony. In future, instead of expensive cellular base stations, service providers may deliver voice using IP protocols over an infrastructure of much cheaper wireless access points (please see Figure I-2). However, because these access points will be deployed much denser, it will be possible to locate a user with a higher spatial accuracy. In addition to anonymous user authentication and billing mechanisms, anonymous routing protocols will be needed that can transfer voice packets between users while offering them location privacy and communication anonymity.

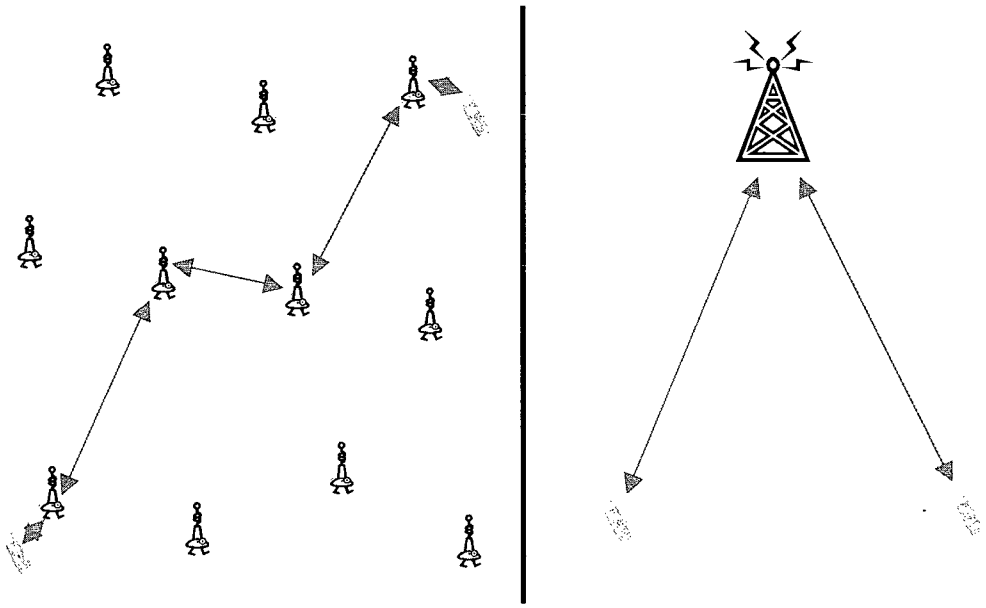


Figure I-2 Wireless services infrastructure

I.3 CONTRIBUTIONS

1. Destination Controlled Routing (DCR)

The main contribution of this thesis is a new paradigm in wireless ad hoc routing in which the destination, as opposed to the source or the intermediary nodes (normally done in existing protocols), is in charge of path selection. In other words, the destination controls how the route from the source node towards it is chosen and configured. Our motivation for this design has been the realization that the most difficult part of establishing an anonymous and location-concealing path for a routing protocol is to hide the identity and the location of the destination from the source, the intermediate nodes and other third parties. Indeed, the most essential piece of information for delivering

packets is *"where the destination is located and how it can be reached"*. Other information such as source identity and its location, QoS parameters, various protocol options and so on help improve the performance of the protocol, e.g. transmission reliability, but are not as important as knowledge about the destination. So, how can we keep the location and the identity of the destination hidden and at the same time provide the possibility to deliver packets to it? Our solution is *Destination Controlled Routing (DCR).* In DCR, the destination instructs other nodes, secretly and anonymously, how to send or forward packets destined for it so that those packets eventually arrive at the destination.

DCR is an abstract concept that may be applied to a multitude of wireless network types. However, due to specific characteristics of each kind of wireless network, it may have to be differently designed in each case. For example, multihop wireless sensor networks (WSN) are a special class of ad hoc networks that are substantially different from generic ad hoc networks in terms of their applications and their properties. Figure I-3 depicts a high level classification of today's wireless technologies.

As can be seen, we have designed the following flavors of DCR:

- *V-routing for wireless mesh networks*
    - *Community Networks* model
    - *Wireless Internet Service Provider (WISP)* model
    - *MANET*
- *DCARPS for multihop wireless sensor networks*
    - with *abundant traffic*
    - with *scarce traffic*

While there have been several proposals for location-concealing and anonymity-preserving schemes based on reactive routing protocols, to the best of our knowledge, our DCR protocols are the first to provide these capabilities in a proactive routing environment.

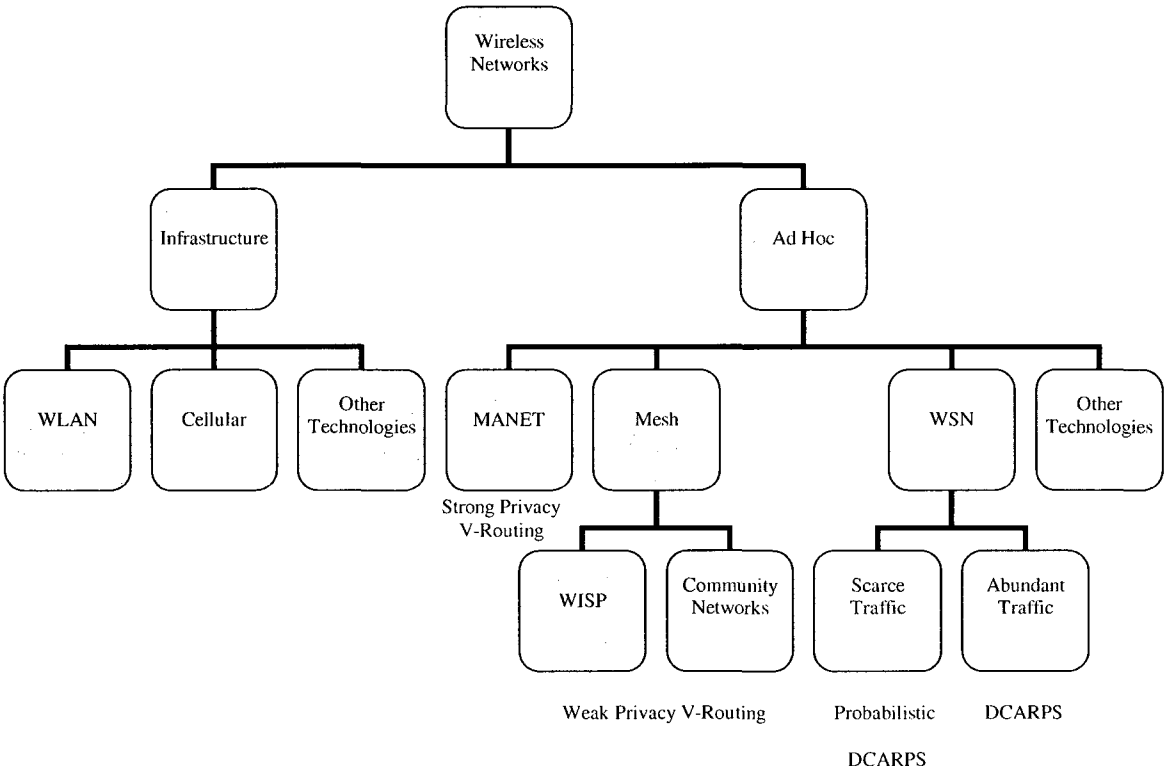


Figure I-3 DCR for various kinds of wireless networks

2. Efficient Network Topology Discovery

In DCR, the destination needs to have a global view of the network topology in order to calculate a path from an arbitrary point to itself. This knowledge is also needed by routers in the V-routing instantiation of DCR, as will be clear later on. One of our contributions is an efficient link state topology discovery protocol called 2hop-clustered NTDP that

significantly reduces the overhead of periodic updates compared to regular link state routing protocols. This new protocol makes link state routing protocols more viable for use in an ad hoc network.

3. Anonymous Network Topology Discovery for WSN

In the case of wireless sensor networks, we also introduce a simpler topology discovery protocol that even though is inferior to 2hop-clustered NTDP in terms of overhead but can protect against network topology disclosure when combined with appropriate cryptographic methods.

4. Beacon-free Topology Discovery for WSN

Another one of our accomplishments is eliminating sink-originated beacons in wireless sensor networks that are commonly used for routing setup in these networks. The sink, sometimes also referred to as the base station, is a special node in sensor networks that is responsible for collecting the data gathered by sensor nodes. In most routing protocols designed for sensor networks such as TinyOS beaconing [32, 33], directed diffusion [34] and minimum cost forwarding [35], the sink periodically broadcasts a beacon that is flooded in the network informing each node of its distance from the sink. Then, based on a simplified distance-vector approach, each node selects an upstream node called a parent in order to forward its data towards the sink. In some other protocols such as rumor routing [36] a similar approach is used whereby the sink broadcasts a query in the network. Although efficient, these methods obviously reveal the location of the sink which is probably the most sensitive piece of information in a hostile environment. Based

on our DCR approach, we allow each node to have a parent node without using sink-originated beacons or queries.

5. Load-balancing Routing Tree for WSN

As part of our DCARPS protocols, we have proposed and evaluated a fair shortest-paths tree construction algorithm that unlike most similar algorithms takes the resource limitations of sensors into account and distributes the children of parent nodes with the same distance from the sink evenly. In other words, while establishing the spanning tree routing tree rooted at the sink, it ensures that the difference between the numbers of children of two neighboring nodes with the same hop-count from the sink is as small as possible. This property is very desirable in a wireless sensor network because it helps drain the energy of all nodes at an even pace and thus prolong the lifetime of the network.

6. An Analytical Tool for Evaluation of Randomized Routing Protocols

In order to evaluate the effectiveness of DCARPS in achieving its objectives, namely source/destination anonymity, we introduced a metric based on entropy. Using this method, our simulations show a good level of performance for this protocol in presence of a global adversary. The success of this protocol was measured in terms of different parameters such as traffic volume, node deployment density and adversary's observation period.

I.2 THESIS ORGANIZATION

The rest of this thesis is organized as follows: in chapter II, we review the related work in the literature and explain our motivations for choosing the proactive routing approach in the design of DCR. In chapter III, we propose our V-routing protocol as an instantiation of DCR for multihop wireless ad hoc networks. In chapter IV, we introduce the DCARPS protocol as an instantiation of DCR for wireless sensor networks with a high traffic volume. In chapter V, we propose probabilistic DCARPS as an extension of DCARPS for sensor networks with a scarce traffic volume. In chapter VI, we describe our 2hop-clustered link-state network topology discovery protocol and evaluate its improvement over regular link-state routing protocols in terms of control overhead. Finally, we conclude with a summary and some suggestions for continuation of this thesis in chapter VII.

I.3 SUMMARY

In this chapter, we offered an introduction to multihop routing in ad hoc networks and the security vulnerabilities of this kind of network. In particular, we explained the importance of location privacy and communication anonymity in these networks and, using some specific example scenarios, demonstrated the shortcomings of regular ad hoc routing protocols in this regard. In order to make our terminology clear, we provided definitions for the anonymous routing notions that we will deal with in this thesis including sender and receiver anonymity, communication anonymity and location privacy. Finally, we gave a brief overview of our own contributions as well as an outline of this thesis.

CHAPTER II

BACKGROUND

In this chapter we provide an overview of the related work in this area of research and point out their strengths and their weaknesses. We have divided the anonymity-related literature into 3 parts corresponding to legacy anonymity systems, anonymous routing protocols for ad hoc networks and anonymity systems for WSN. In the last section, we have provided a comparison of proactive and reactive routing approaches.

II.1 LEGACY ANONYMITY SYSTEMS

In the Internet, proxies are usually used to provide anonymity for web transactions. A proxy is a third entity that is inserted between two communicating entities in order to hide the identity of the sender from the receiver. The goal of this approach is to hide the identity of a web user from the web servers. Lucent's Personalized Web Assistant [37] is built using this approach. A *mix* [1] is an enhanced proxy that in addition to sender anonymity provides communication anonymity against a global eavesdropper in a network. It collects messages from different senders, reorders them, cryptographically alters them, pads them to equal lengths and then forwards them toward the receiver. A sequence of mixes can make it difficult for an eavesdropper to determine which message came from which sender. Mixes provide receiver anonymity when an adversary knows the source.

Anonymity schemes that are designed for the Internet (such as mix-net and many variations of it) are not suitable for mobile ad hoc wireless environments for many

reasons e.g. lack of infrastructure, resource limitations and dynamic topology. For example, they assume a static network of special routers (even in onion routing [38], the onion routers are connected with longstanding socket connections) and employ excessive padding and cryptographic methods. They are mainly application-layer protocols that are designed with the client-server model of the Internet in mind. Therefore, they are usually sender-centric meaning they are mostly concerned with the security of the source of a transaction. Numerous flavors of these systems have been designed for mobile networks (e.g. [39], SDAR [40]) but they are still mostly sender-oriented. Two exceptions are ANODR [41] and MASK [42], which support destination security too and will be discussed later.

Another issue with these systems, including those versions developed for wireless networks, is that they use an on-demand approach meaning they establish a path between the sender and the recipient when such a path is needed and discard it afterwards. This approach, as opposed to the proactive approach that maintains routing information all the time, has efficiency advantages but instead creates a problem usually referred to as *slow start* that will be explained later in the next chapter.

In cellular systems, more work has been done on providing location privacy for users compared to Internet protocols. For example, in [43] Reed et al use anonymous connections as primitives to describe protocols that provide location-protected communication over public network infrastructure. Specifically, they briefly describe a method for indirectly connecting two mobile phones without revealing their locations, based on a paging system and call back numbers. However, this and other anonymity and location privacy schemes that are designed for these systems (e.g. [44, 45]) are not

suitable for ad hoc networks either, since they too rely on a stable network infrastructure. Like many of the systems designed for the Internet, they rely on constant availability of special on-line services such as look-up directories. Last but not least, they do not address particular location privacy issues of ad hoc networks explained in the previous section.

II.2 ANONYMOUS ROUTING FOR AD HOC NETWORKS

Most of existing secure routing protocols for ad hoc networks (e.g. [46-54]) have mainly focused on ensuring the authenticity and integrity of routing messages in order to provide resilience to tamper, eavesdropping and Denial of Service (DoS) attacks by malicious entities. There are a few that have considered source node privacy and security in some cases (e.g. SDAR [40], AnonDSR [55]). However, to the best of our knowledge the only proposals that have also attempted to preserve the privacy of the destination node in terms of identity and location are ANODR [41] and MASK [42] which are very similar and will be reviewed in this chapter. Moreover, these protocols provide these services only with regards to external adversaries and perhaps the network itself. To the best of our knowledge, ANODR and MASK are the only proposed schemes that have tried to hide the location of the destination from the source and vice versa, in addition to having other security properties. But unfortunately, virtually all of these efforts have been directed towards reactive routing protocols.

ANODR

ANODR [41] is an on-demand routing protocol designed for operating in hostile environments. It supports communication anonymity as well as location privacy and

identity anonymity for sender, receiver and nodes en route in presence of eavesdroppers and intruders (compromised nodes). ANODR is similar to AODV but uses a label-switching technique with pseudonyms for every hop of the path from the source to the destination. An attacker eavesdropping on the local transmissions is unable to infer from where the packets are coming and to where they are going. This protocol uses identity pseudonyms for nodes that are unlinkable to their locations. This way, it prevents an attacker from tracking a multihop route from a starting point to other nodes on the path.

In order to reduce the cost and latency of its cryptographic onion approach, ANODR uses a technique called Trapdoor Boomerang Onion in the route discovery process that makes sure no local or global eavesdropper can learn the complete path. However, ANODR has several practical issues including:

a. It relies on the existence of a global trapdoor that implies the source and destination have a pre-established shared secret. This assumption is not realistic in most scenarios. Besides, since the ID of the source is hidden until the trapdoor is open, the destination must keep trying all of the secret keys that it may share with other nodes before it can successfully decrypt the route request message from a particular source node.

b. In the route acquisition phase, each intermediary node performs cryptographic operations in both the forward and the reverse directions. These operations add significantly to the path setup delay from which a reactive routing protocol already suffers, as will be explained in the next section.

c. In order to thwart traffic analysis attacks that attempt to correlate RREQ (Route REQuest) and RREP (Route REPly) packets and therefore trace the route, ANODR uses asymmetric cryptography at each node to encrypt some fields in the RREQ and RREP messages that do not change over different hops. In order to achieve this, ANODR needs an asymmetric secret channel between the producer of an RREP message and its receiver. To establish such asymmetric channel, during the RREQ phase, each forwarder transmits its public key to the next hop. In the RREP phase, the producer of the RREP message uses this key to encrypt an unchanging field called "proof of trapdoor opening". This process requires memory for key storage at each node and incurs overhead for key transmission. It also requires processing power for key generation, and encryption/decryption operations. Moreover, it further increases the path setup latency.

d. The route maintenance protocol of ANODR forces the source to start the route discovery process from the beginning every time the path is broken, for example due to mobility. Besides, such an event can only be detected when an ACK from the destination node fails to arrive at the node before a broken link, which is the node that will generate a RERR (Route ERRor) message. On the other hand, the authors assume that an intermediate node can detect a broken link but they do not specify how this capability is implemented. Specially, reliability in the transmission of RERR and RREP messages depends on such a capability.

a. On-demand routing protocols such as DSR and AODV rely on various optimization techniques to reduce their considerable overhead. However, secure and anonymous routing protocols including ANODR are not able to take advantage of these techniques because they violate security and anonymity objectives. For example, some of these techniques are used to repair routes that are broken due to mobility as well as allow for faster route replies by intermediate nodes that happen to have a route to the desired destination. But these schemes are in conflict with the anonymity goal because they need nodes to cache routing data that are not relevant to the route currently being used.

MASK

MASK [42] is another identity-free anonymous routing protocol very similar to ANODR except that it takes a different approach to generating route pseudonyms. It uses an anonymous neighbor detection mechanism that allows any two neighboring nodes to mutually authenticate each other in order to ensure that they both belong to the same party. During this process, each two legitimate neighbors derive several pairs of <shared secret key, linkID>, where the linkID is a random link pseudonym similar to ANODR's route pseudonyms. These pairs will be used one per link transmission and are replaced frequently, which means more processing power is needed to regenerate them. MASK assumes that an offline trusted authority generates and allocates a big set of pseudonyms to each network member. Each node must store this set as well as one set of secret pairs per each neighbor.

Mist Routing

This routing protocol, proposed in [56], combines handle-based hop-by-hop routing with PKI in order to provide anonymous communication. It requires a hierarchy of special routers called a *mist* to be installed that performs handle-based routing to hide information about the location of the initiator and the responder, even from each other. This protocol allows the system to use sensors that can detect the presence of users in a room without the ability to identify the users. The mist allows a user to authenticate him/herself to the system without revealing its location. Each user is always connected to one of the mist routers (called its *lighthouse*) but no one even the lighthouse knows where exactly s/he is located.

Mist routing keeps the location of the destination hidden from the mist routers and the sender. Leaf nodes in the mist hierarchy have knowledge of the location of the user but not its identity. Leaf nodes refer to users using handles, which are pseudonyms. Lighthouses have knowledge of the users' identities but not their locations. To send a packet, a user must establish a connection to its lighthouse and then the packet is transmitted to the destination lighthouse and eventually to the destination node. The process of establishing a "virtual circuit" between the source and its lighthouse is called handle-based hop-by-hop routing, which is similar to label switching. To send a packet to a user, other users need to locate its lighthouse first. Two centralized approaches are used, LDAP and web servers.

Mist routing is aimed at infrastructure-based networks owned by organizations intending to provide mobility to their users while still being able to always authenticate them without compromising their anonymity and privacy. In other words, it prevents a

ubiquitous computing system from turning into a surveillance system, the so-called Big Brother problem. To achieve this objective, it relies on a fixed logical hierarchy of special routers and sensors (called portals) with pre-determined ancestral relationships. Portals are fixed sensors installed in pre-specified locations that can sense the existence of a user but cannot identify him/her. This approach requires a fixed network structure and is unsuitable for a dynamically changing ad hoc network.

SDAR

Secure Distributed Anonymous Routing (SDAR) [40, 57] is a protocol based on onion routing that adds security to the route discovery procedure of on-demand routing protocols. When this message is broadcasted by the source, only those intermediary routers that satisfy certain trust requirements specified by the source can be added to the path. They insert their ID and a session key to the message and forward it to the next hop until it reaches the destination. This information is encrypted before being added. When the message is returned, the source has all the information about trustworthy nodes on the path and the keys to encrypt for them.

SDAR is very similar to SAR [52] in that both find routers that satisfy certain security requirements. It is obvious that this protocol does not hide the location of the destination from the source and does not provide communication anonymity.

AO2P

AO2P [58] is a position-based on-demand anonymous routing protocol that offers communication anonymity. It delivers packets to a geographical location where

destination has been reported to reside lately and does not use node identities. Several problems can be seen with this protocol. For example, the premise of this protocol is that only one node exists at any particular position at any time. Nodes must be equipped with GPS, several special position servers are needed and the position management system produces additional overhead in the network. Besides, the source can legitimately learn the exact location of the destination node, eavesdroppers can trace the RREQ packet to the destination using an un-mutable field in it called "authentication code" and they can trace the RREP packet back to the source by correlating the RREQ and RREP packets. Moreover, the rate of packet delivery failure for AO2P is expected to be high because of destination mobility. The destination may have already moved by the time the packet reaches its previously reported position. This causes repeated route discovery, which means more overhead.

AO2P uses a 4-pass route discovery protocol that incurs a lot of latency. Also, the node pseudonyms and temporary MAC addresses used in the route discovery messages in AO2P are generated by a hash function, which takes only the current time and the source node's position as inputs. An adversary that learns this hash function becomes very powerful. For instance, the source does not seem to have to authenticate itself to the destination.

AnonDSR

AnonDSR [55], which is an anonymous routing protocol based on DSR prevents a data packet from being traced back to the sender or receiver but this protection is limited to the data transmission phase. In the route acquisition phase, similar to regular

DSR the identities of the two end nodes and all the intermediary nodes are transmitted as clear text. Moreover, the RREQ packet carries a temporary public key that is fixed across all the hops between the source and destination. An omni-present adversary (a global adversary who can monitor all transmissions) can use this field to trace the packet back to the source and the destination.

PPR

The Privacy-Preserving Routing (PPR) protocol presented in [59] uses a combination of proactive and reactive approaches. This protocol is designed for a hybrid multihop wireless network consisting of base stations and mobile nodes. Each base station covers an area serving many mobile nodes that only communicate with the base station. However, mobile nodes do relay packets for other nodes and the base station in their coverage area. PPR consists of two parts; a down link protocol and an uplink protocol. The downlink part is actually a source-routing protocol similar to DSR, which assumes that the base station knows the complete topology of its control area. In the uplink, each node only forwards the packets towards the base station through the link that has the smallest hop count to get to the base station. To acquire the network topology in its control area, the base station periodically initiates a secure topology discovery procedure similar to DSDV. During this process, all nodes must anonymously calculate their neighborhood information and send it to the base station in an encrypted format.

SERAN

All of the above-mentioned protocols take a reactive approach to routing. As we pointed out before, aside from other shortcomings of each particular protocol, this approach

results in a long path-establishment delay. One of the few privacy-preserving solutions for networks that use proactive routing protocols is SERAN [60]. However, from the description found in [60] the problem statement is not very clear. It seems that the idea is to protect/isolate a node from adversaries by revoking its IP-address and only giving it one when it needs to send or receive packets. To implement this function, a smart card is used that runs DHCP in order to assign an IP address to the node on demand although it also seems that a fixed IP address is programmed in the smart card. The authors argue that this way the node would be invisible to attackers. When another node needs to communicate with the isolated node they will go through a handshaking process whereby they exchange IP-addresses. However, it is not clear how mere knowledge of a node's IP address is enough to deliver packets to it when this address is not present in any routing table.

Table II-1 compares some of the most important anonymous routing protocols for multihop ad hoc networks including our own V-routing.

Several more anonymous routing protocols can be found in [61-72]. All of these protocols are based on reactive routing and some are variants of the protocols introduced above. In particular, authors of ANODR have shown that ASR [61] is a variant of this protocol. ARM [62] strives to resolve some of the problems of other protocols such as ANODR, MASK and ASR but still relies on an apriori association between the source and the destination nodes. PRISM [63] is a location-based protocol where every node is assumed to have a GPS and thus it suffers from problems similar to AO2P. [64] tries to reduce the delays associate with anonymous routing protocols by using only symmetric cryptography. [65] reduces the use of public key operations by employing a hierarchical

architecture in ad hoc networks. [66] is yet another attempt at adapting mix networks to a dynamic ad hoc network topology. The protocol proposed in [67] uses a callback service between the source and the destination nodes and also requires a secure out-of-band channel for them to share a secret. A scheme similar to onion routing but without the use of PKI is proposed in [68]. ODAR [69] tries to enhance the performance of on-demand anonymous routing using Bloom filters. A multicast version of ODAR is introduced in [70]. Two more multicast anonymous routing protocols are proposed in [71, 72]. The protocol described in [72] is based on underlying unicast on-demand anonymous routing protocols such as ANODR but the source node (multicast root) must know the locations of destination nodes (multicast subscribers).

Table II-1 V-routing vs. existing anonymous routing protocols

Protocol	Routing	Infrastructure	Security Shortcomings	Comments
V-routing	Proactive	No		Triangular routing
ANODR	Reactive	No		- Needs pre-established secret between source and destination - Impractical trap door - Slow and expensive route repair - Heavy cryptography at intermediate nodes
MASK	Reactive	No		High memory and CPU requirements at intermediate nodes
MIST	Proactive	Yes *portals, special routers, LDAP and web servers*		Unsuitable for ad hoc networks
SDAR	Reactive	No	- S learns D's location - No communication anonymity	
AO2P	Position-based Reactive	Yes *special position servers*	- S learns D's location - Adversary can trace RREQs to D and RREPs to S.	- Needs GPS. - Assumes only one node at a position. - Overhead of position management system. - Data delivery failure if destination moves.
AnonDSR	Reactive	No	Protection limited to the data transmission phase	
PPR	Reactive & Proactive	Yes	- BS know the locations of all nodes - Every one knows who and where the BS is. - No comm.. anonymity with regards to BS.	Triangular routing

II.3 ANONYMITY SYSTEMS FOR WIRELESS SENSOR NETWORKS

Phantom routing [73, 74] is one of the earliest works that clearly addresses the problem of source location privacy in wireless sensor networks. It assumes that movements of an

object are continuously monitored by a network of stationary sensors and an adversary tries to locate the object by tracing the stream of packets flowing from its point of presence towards the sink. In order to mislead the adversary, phantom routing first sends each packet from the source randomly to an intermediary node called a phantom source located a few hops away from the real source. This process is called a random walk. The phantom source either floods the received packet in the network or unicasts it to the sink.

Obviously, the flooding version of this protocol does not reveal the location of the destination but it has the disadvantages of flooding including excessive energy and bandwidth consumption. On the other hand, the single-path version cannot hide the location of the sink as it uses regular single-path routing schemes that require all sensors to know where in the network the sink resides. Moreover, the random walk phase of phantom routing threatens source-location privacy. In this phase, the packet carries a hop-count that is initially set to zero and then incremented at every hop. When the hop-count reaches a pre-specified value, the packet is flooded. This implies that an eavesdropper who happens to intercept a packet during its random walk can find out how many hops it is away from the source. When multiple cooperating adversaries combine such information they can narrow down their search for the source by finding intersections of their estimated areas for the source location. A simple solution to rectify this problem would be for the source to initialize the hop-count to a random value for each packet instead of zero. Then, each forwarding node would decrement this value until it reached zero, at which point the packet would be flooded. This way, an adversary could not guess how far away it is from the source. It could only guess in how many retransmissions the packet would be flooded.

Another issue with phantom routing is user anonymity. If more than one sink exist in the network, both versions of phantom routing require each packet to clearly specify its destination. Therefore, destination anonymity and communication anonymity are not preserved by phantom routing if no modification to this protocol is considered.

The work in [75] improves upon Phantom routing by using a two-way random walk. [76] and [77] use fake messages in order to mislead the eavesdropper and thus protect the location of the source node.

Hong et al. [78] introduce two algorithms to protect sensor networks against timing analysis attacks. As mentioned in the previous chapter, timing analysis is a kind of traffic analysis that exploits timing correlation of transmissions from neighboring nodes to correlate packets on successive links and therefore uncover end-to-end traffic flows between the source and the destination of those packets. Most of the traditional countermeasures against this kind of attack, such as packet reordering and decoy traffic, are not suitable for the resource-constrained sensor networks. The authors base their algorithms on adding random delays to packet retransmissions at each forwarding node, in an attempt to obfuscate the temporal relationship among packet transmissions in a neighborhood. This paper is not concerned with the actual routing of packets. In other words, it does not specify how a path for a packet is found towards the sink.

Another kind of traffic analysis attack tries to gain sensitive information about the end-to-end data flows by monitoring the changes in the link-level traffic patterns. The heuristic algorithm proposed in [79] tries to prevent this type of attack by routing end-to-end data flows in a way that keeps the global view of link-level traffic patterns on all the links across the network as invariable as possible. In fact, when the real traffic pattern in

the network changes, this algorithm reroutes traffic flows in a manner that the overall network traffic patterns remain unchanged. However, this paper does not discuss implementation issues of this algorithm and their privacy implications. This algorithm requires complete knowledge of link-level and end-to-end flows throughout the network. In a centralized implementation of this algorithm for sensor networks, the sink would be the sensible choice as the entity performing the algorithm. In that case, protocols are needed for conveying necessary information from the sensors to the sink, as well as procedures for setting up the right link-level connections by the sink, all in a secure and private manner. A distributed implementation is more challenging, especially from a security standpoint. It would mean that global knowledge of data flows and network topology must be available to all sensors.

The focus of the work reported in [80] is protection against intrusion attacks and traffic analysis attacks that aim at isolating or locating the sink. It employs two main techniques to prevent such attacks. First, it increases tolerance against sink isolation by using redundant sinks and develops secure multipath–capable path setup mechanisms, so that sensors can report their data to multiple sinks. However, this path setup mechanism is based on sink-originated beacon signals, which reveal the location and identity of the sink, as we will explain later in this chapter. It also needs all nodes to identify their neighbors, which may also be a privacy concern. The second countermeasure offered in this paper is the use of anti-timing analysis techniques that prevent an attacker from tracking packets back to the sink by monitoring the transmission times of a sensor and its parent(s). It achieves this goal by unifying the transmission rates of all nodes, using delays and dummy packets, when necessary. This technique suffers from problems such

as waste of energy and bandwidth, data loss due to buffer overflows and latency due to random delays.

Deng et al. [81] use four traffic randomization techniques to protect against traffic analysis attacks that aim at locating the sink in a WSN. They try to increase randomness in traffic patterns, in order to confuse an attacker who exploits pronounced traffic patterns due to fixed-path routing protocols. Traffic analysis attacks of the "rate monitoring" and "time correlation" types have been considered in this paper. The first technique proposed in this work is a per packet random multiple-path forwarding scheme used at each node. The second technique is a controlled random walk that uses a probabilistic forwarding scheme at each node to determine the next hop according to a uniform probability distribution, in order to protect against rate monitoring attacks. Using these multi-parent routing and random walk techniques, this proposal spreads a data flow over many end-to-end paths and as such can be useful in hiding the physical locations of the source and the destination against path tracing attacks. However, this scheme is based on the common topology discovery method of propagating beacons from the sink, which as mentioned before, reveals the location of the sink. The last two techniques use fake paths and fake hotspots that are based on decoy traffic, which as mentioned before, consumes energy and bandwidth as well as degrades performance in terms of packet delivery success ratio and latency due to increased rate of collisions.

Olariu et al [82-85] intend to hide the identities and locations of the source and destination in a wireless sensor network, as well as provide traffic untraceability by imposing an anonymous virtual infrastructure over the physical infrastructure. They define a training protocol that organizes nodes into clusters that are created from the

intersections of virtual wedges and concentric circles. This process provides a polar coordinate system that gives nodes a sense of their locations relative to the origin of the coordinate system, without assigning network IDs to nodes. As far as a routing protocol is concerned, if the sink is the center of the coordinate system, each packet is simply routed along the same wedge where it is originated, using multihop routing among clusterheads. In that case, it is obvious that an eavesdropper can locate the source and the destination of the packets by simply following the links. When the sink is not at the center, the details of the routing protocol are not provided.

II.4 PROACTIVE VS. REACTIVE ROUTING

In this section, we explain our motivations for choosing the proactive routing approach for DCR over the reactive approach. Despite their long successful history in wired networks, proactive routing protocols (e.g. OSPF [86] and RIP [87]) proved at first to be inefficient in ad hoc networks. This was mostly due to their large control overhead generated by periodic routing updates needed to keep nodes' routing tables correct at all times. In frequently changing topologies of ad hoc networks, these updates have to be broadcasted more often, which means more consumption of power and bandwidth. Another problem with proactive routing protocols was their memory requirements in order to store routing tables on each node containing routes to every possible destination. Because of these reasons, the new reactive routing protocols designed for ad hoc networks (e.g. DSR [17, 18] and AODV [20, 21]) proved to be more efficient and scalable than their proactive counterparts (e.g. DSDV [26]). These protocols only create routes when they are needed and discard them when they are no longer used.

Because of the success of reactive routing protocols in ad hoc networks, almost all of the efforts in the field of anonymous routing for this kind of network were also focused on this class of routing protocols. ANODR [41], MASK [42] and SDAR [40] are some examples. However, due to advances in radio technology on one hand and design of improved proactive routing protocols on the other, the outlook is slowly changing. Bandwidths of upwards of 100 Mbps [88] are now available in wireless networks, which means larger amounts of routing updates can be accommodated. Also, mobile devices are nowadays equipped with much more usable memory.

New proactive routing protocols for ad hoc networks (mostly based on the link state routing approach) have been designed that reduce the amount of routing overhead significantly via efficient dissemination techniques. Among these protocols, OLSR [28] and TBRPF [29] are now two of the three MANET RFCs in the area of routing. Furthermore, in emerging network architectures such as mesh networks [89-92] and wireless hybrid networks [59, 83- 85] a backbone of low-mobility nodes usually exists. These nodes that normally act as access points for mobile clients can efficiently use proactive routing protocols because of their fairly static network configuration. The next section provides more information about mesh networks. TBRPF is commercially used in the Firetide brand mesh network routers [91] while OLSR is also specially designed for mesh networks. In the iMesh infrastructure described in [89], OLSR has been chosen as the routing protocol. Also, in the MeshCluster network architecture described in [90], OLSR is considered a good choice when the traffic is uniformly distributed across the network.

STAR [16] is another proactive routing protocol that uses efficient dissemination techniques. It claims to have even less overhead than reactive routing protocols. Other proactive protocols proposed recently for ad hoc networks include Hierarchical Link State [31], FSR and HSR [14]. The last three protocols use limited dissemination techniques and hierarchical organization to reduce the overhead of periodic routing updates. In certain network scenarios, even traditional proactive routing protocols designed for wired networks such as OSPF may be preferable. Another advantage of such proactive routing protocols is that they have been successfully in use for a long time and have the backing of much experience.

The choice of a routing protocol for a certain network is influenced by many factors such as network size, node mobility, resources, overhead constraints, complexity, supported applications, etc. One of the main drawbacks of reactive routing protocols is the *path setup delay* that they incur since they only begin finding available routes when the source node needs a connection. This problem also known as *slow start* is considered an important disadvantage when interactive applications such as multimedia communications and web surfing are concerned. Proactive routing protocols tradeoff routing update overhead for constant availability of routes and at the moment they are needed. Besides, as mentioned before, several techniques have been proposed that reduce the amount of processing and communication overhead of these routing protocols. The latency of reactive protocols in finding routes in the face of link failures also translates into dropped packets while this is much easier to avoid in proactive protocols as alternate routes are immediately available.

One of the rare experimental studies of ad hoc networks [97] has compared AODV and OLSR. These experiments have shown that the slow start problem of AODV also causes problems for higher layer protocols specially TCP mainly due to connection timeouts. It has also shown that although the overhead of OLSR is higher than AODV it is still quite small even compared to the available bandwidth of IEEE 802.11b. In chapter VI, we propose a new method that reduces the topology discovery overhead of proactive routing protocols by more than 50% through the use of the clustering technique in order to eliminate the redundancy in routing update messages.

Looking at the performance of a protocol from the point of view of control overhead is a simplistic perspective. The total overhead of a protocol is the result of not only control traffic but also sub-optimal routing [98]. Reactive routing protocols consume extra bandwidth (a form of overhead) and introduce extra delay due to routing sub-optimality because they either do not try to find the best routes or try to use the established routes as long as they are available even though they may no longer be the best routes. Proactive routing protocols that do not use limited dissemination techniques can provide optimal routes since they use full knowledge of network topology. That is also why they are better capable of supporting QoS because they can know the quality of the links in terms of delay, BW, etc before establishing a path [14].

Ultimately, no routing protocol can be considered the best protocol for all possible scenarios. Some researchers go as far as suggesting that the routing protocol must be dynamically chosen for the current network conditions such as network load, mobility, link characteristics, QoS requirements and security concerns. This is the premise of the ARRCANE project supported by Ericsson and NUTEK [99]. Also, in [90]

switching between AODV and OLSR has been considered a possibility based on the traffic patterns. Since proactive routing protocols like DSDV are known to work well with a small number of nodes while reactive routing protocols are efficient in small geographic areas, several protocols have been proposed that combine these two groups of protocols in a group called *hybrid* routing protocols e.g ZRP [27].

The above arguments show that the future ad hoc networks will not be solely relying on reactive routing protocols but proactive routing protocols will also exist either independently or in conjunction with reactive protocols. Yet, virtually no attention has been paid to developing methods for protecting location privacy and anonymity for the proactive group of routing protocols. Our DCR approach is an answer to this problem.

II.5 SUMMARY

In this chapter, we had an overview of the main literature in the area of anonymous routing. First, we explained why the traditional anonymity systems are not applicable to ad hoc networks. Then, we considered recent protocols that have been designed for ad hoc networks and sensor networks and pointed out their shortcomings as well as their capabilities. We also briefly mentioned the two big differences between our DCR approach and most of these protocols i.e. being destination-centric and being based on proactive routing. Finally, we provided some justifications for choosing the proactive approach for DCR vs. the reactive approach for routing.

CHAPTER III

V-ROUTING

In this chapter, we apply DCR to multihop wireless ad hoc networks. First, we describe two versions of V-routing for mesh-like networks; one used with a wireless community network model and another used with a WISP (Wireless Internet Service Provider) network model. These two protocols provide what we call weak privacy because they do not provide the specified security properties with regards to all existing third parties. Then, we propose a version of V-routing for MANET networks which provides strong privacy as opposed to weak privacy. We Also propose a public-key certificate management system for V-routing, analyze the performance of this protocol in terms of overhead and security and explain how V-routing can be used to support anonymous multicast applications,

---

In this chapter, we propose a family of new anonymous routing protocols called ***V-Routing*** as instantiations of DCR. In all of the three flavors of V-routing described in this chapter, a user node intending to receive packets from future connections establishes a secret path from a remote router located in the network (which we name a *rendezvous point*) to itself, implemented by passing through multiple hidden virtual hops. Potential senders submit their packets destined for that user to its rendezvous point first, which will then be forwarded over the secret path. Using this indirect end-to-end path formed by 2 segments and hop by hop routing, V-routing allows a mobile node to send and receive

packets anonymously while keeping its location hidden and also maintaining communication anonymity from third party observers.

III.1 OVERVIEW

In the spirit of taking the proactive approach to routing, in order to avoid the path setup latency of reactive protocols, the V-Routing protocol works in a proactive manner. It consists of three parts; *path establishment, data transmission* and *path termination.* Before a node that requires location privacy can receive data packets, it has to go through the path establishment phase and set up at least one secret route towards itself, starting at another arbitrary node (its rendezvous point). In other words, the main part of routing is in the control of the destination; that is the key design element for achieving destination location privacy in V-routing.

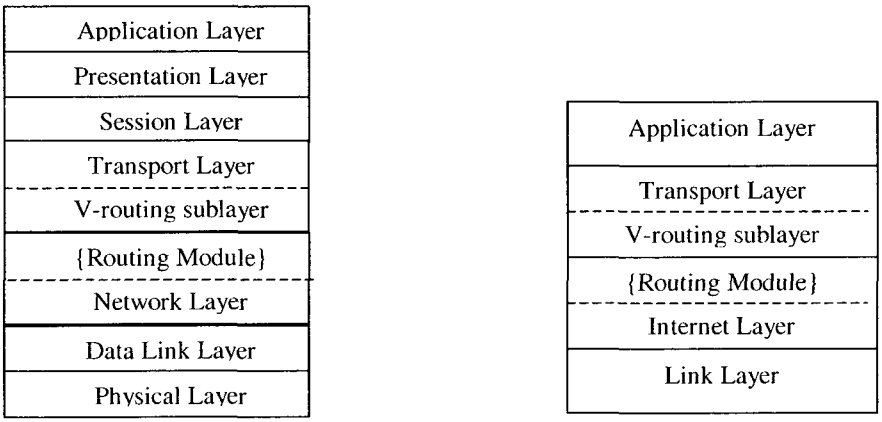


Figure III-1 V-routing in the (a) OSI and (b) TCP/IP protocol reference models

As shown in Figure III-1, V-routing is an overlay protocol in the sense that it uses the services of an underlying proactive routing protocol that is responsible for delivering packets. It allows the destination to establish a secret virtual path on top of the actual route in a way that its location is hidden even though it remains reachable. Therefore, unlike most of the existing anonymity systems, it is able to support destination location privacy.

In the next section, we will describe our assumed adversary model which explains the capabilities of the attacker trying to violate the privacy rights of the user. An important issue in the design of a security protocol is the notion of *trust* and who is *trust-worthy*. In our case, node A is considered trust-worthy by node B if node B is willing to share a secret with A. A secret may be one's identity or location or associations, etc. In section III.4.2, we introduce two versions of V-routing (one for wireless community networks and the other for a WISP network) for situations where the destination trusts at least one other node in the network. We refer to this design as *V-routing with weak privacy*. In section III.5, we describe a different version of this protocol, called *V-routing with strong privacy*, for situations where the destination has no trust relationship with other nodes. In each case, we will specify the degree of trust between a user and special nodes in the network such as its access router, the rendezvous point, peer nodes, etc. in the form of a privacy objective statement.

In section III.6, we explain in details why V-routing achieves its security objectives. In section III.7, we discuss some performance issues of V-routing such as computational and communication overhead. In section III.8, the key management and

certificate management systems- to be used with V-routing- are proposed and in III.9, a new anonymous multicast protocol that is based on V-routing is presented.

III.2 ADVERSARY MODEL

We assume the existence of an omni-present external adversary sometimes called a *global adversary* that can monitor all transmissions throughout the network. The adversary is assumed to be very strong in terms of processing power and storage capacity but it cannot break the cryptographic measures used by legitimate nodes in bounded time. It launches only passive attacks (eavesdropping) not active attacks such as DoS (Denial of Service), impersonation, message modification, man-in-the-middle, etc. These attacks can be discovered and the culprit can be identified using intrusion detection techniques. The kind of adversary that we have assumed wishes to remain undetected until it gathers its desired information.

The adversary can compromise a node and become an internal attacker. In that case, all of the information stored in the compromised node including the routing information and the cryptographic material such as its shared keys and private keys are assumed to be available to the attacker. However, the adversary can only compromise a fraction of nodes. As far as the location privacy of a node is concerned, all other network nodes even though legitimate, are considered internal adversaries. As far as communication anonymity is concerned, all internal nodes other than the source and the destination are considered internal adversaries. An internal adversary is assumed to be honest but curious meaning it follows all network protocols but tries to passively gain unauthorized access to sensitive information.

We assume that the adversary is capable of performing traffic analysis. Attackers use traffic analysis techniques including content analysis and timing analysis to gain meaningful knowledge about the data flows, even though the data itself may be encrypted. They can exploit things such as packet length, packet headers, timing of transmissions on successive links, traffic patterns and so on to infer valuable conclusions about communications in the network. While we do not concern ourselves with traffic analysis too much, we do address one aspect of *content analysis* that is related to routing. A global adversary can match certain fields in packets (that either do not change or change in a predictable fashion) across successive links in order to trace them back to the source and/or the destination. For example, an RREQ packet in AODV can be traced using its sequence number. To resist this kind of threat, all packets must look completely independent of each other on different links, a notion sometimes referred to as *one-time packets*.

Two methods are in common use for achieving this effect; one is to use onion cryptography where a new layer of encryption is applied to the packet for every intermediate hop, thus changing how it looks on consecutive links. Usually, the source node encrypts a message with the key that it shares with the destination or with the destination's public key. The next-hop forwarder, encrypts the received packet again for the destination, so on and so forth. The destination, applies sequential decryption in the reverse order until it recovers the original message. This method requires either a PKI infrastructure or pre-established shared keys between the destination and all the intermediary nodes. Alternatively, the source could wrap the message in layers of encryption using the keys that it shares with the intermediate nodes (or with their public

keys). Then, each forwarder node would decrypt one layer of the packet (this is the method used in Onion Routing [38]). The second method is per-hop encryption/decryption (re-encryption) between forwarding nodes. For example, in IEEE 802.11, the WEP (Wired Equivalent Privacy) key option allows neighboring nodes to share a link layer secret key [100]. On the higher layers of the protocol stack, each two consecutive forwarders (with a physical or a logical link) can use public key cryptography or shared secret keys to achieve a similar effect. In our protocol, we have used the second method (per-hop encryption) to prevent content analysis attacks.

III.3 V-ROUTING WITH WEAK PRIVACY

This is the case where the destination trusts at least one other node in the network. This – trusted- node may be its immediate point of access to the network or it may be a forwarding node similar to a home agent in Mobile IP [101]. We refer to this design as *V-routing with weak privacy*. In section III.5, we introduce the *V-routing with strong privacy* protocol for situations where a node has no trust relationship with other nodes.

V-routing with weak privacy provides the following services:

1. Hides the location of the destination node (*D*) from the source node (*S*) and all other nodes (internal and external entities) including those on the source-destination path except the one router with which the destination node is associated, namely *AD* (Access router of D).

2. Hides the location of the source from the destination and all other nodes including those on the source-destination path except its home router, namely *AS* (Access router of S).

3. Hides the fact that the source and the destination are communicating from every one (both internal and external entities). Even the two access routers cannot know both ends of the connection.

The terminology used above becomes clear in the following section.

III.3.1 Network Model

One of our main assumptions with regards to V-routing with weak privacy is that at any given time there are a number of legitimate member nodes in the network, which do not require location privacy and anonymity. Therefore, these nodes participate fully in the routing process i.e. they identify themselves to their neighbors, collect and broadcast their neighborhood information (about neighbors that authorize it) to other nodes by way of routing updates and forward packets for other nodes according to their own routing tables. We refer to these nodes as *routers* or *access points.* On the other hand, some of the nodes in the network may like to hide their locations and their movements as much as possible. We refer to these nodes as *ordinary nodes* or *clients.* If a client does not need location privacy, it may broadcast its neighborhood information but it would still identify itself as a client meaning it would not act as a router[3]. This way, it can avoid the costs

---

[3] We will explain later that this design is tailored towards 1-hop clustered architectures, such as infrastructure mesh networks. A different version of our protocol, not described in this thesis, can be designed for networks such as 4G in which user devices belong to multihop clusters and participate in packet forwarding within their cluster. An example of this kind of networks is a multihop WiFi hot-spot also known as microcell technology.

associated with the location privacy protocol. In other words, our protocols give the client the option to dynamically decide whether it wants to remain hidden and pay the cost of privacy or prefers to bypass our security mechanisms and reveal its location.

Every wireless device is identified by a unique location-independent network identifier (ID). In small ad hoc networks such as enterprise or university campuses, private IP addresses are usually used that are all in the same range and do not reveal the location of the user. V-routing is specially intended for this class of networks. In bigger networks, a location-independent global identifier such as a Uniform Resource Locator (URL) may be used. A DNS may translate a URL to an address that is location dependent but the revealed location in this case is usually a big area. Ultimately, we believe that the naming system is out of the scope of V-routing. This protocol prevents the routing protocols from revealing the location of a node as long as the ID of the node does not readily reveal its location.

Figure III-2 illustrates the network architecture in which we are interested. We call a neighboring router of a client with which the client associates an *Access Router* of that client. In this thesis, we consider only one access router per ordinary node but an extension of our protocol to support multiple access routers is straight-forward. The access router of the source node (S) is denoted by AS and the access router of the destination (D) is denoted by AD. In its routing update messages, an access router does not advertise the IDs of those of its members who require location privacy but it advertises those members that do not specify this requirement. A client connects to its access router on the link layer without specifying its own MAC address. Instead, it encrypts its ID with the secret link-layer key that it shares with its access router.

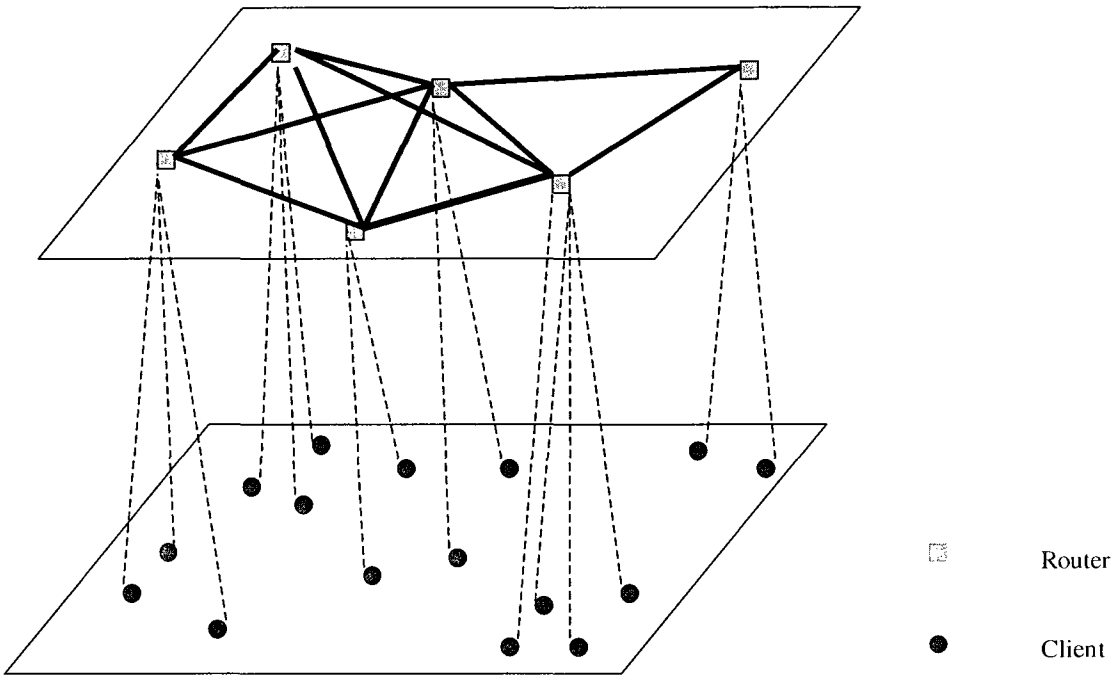


Figure III-2 V-routing network model

Most link-layer protocols such as IEEE 802.11 support security and allow the two ends of a link to share a secret encryption key.

This network model is essentially a clustered architecture with access points as clusterheads which are directly connected to their clients. An example, which is akin to our assumed network structure, is an *infrastructure wireless mesh* [102] network in which clients do not participate in routing. The strong privacy version of V-routing explained later supports a more general model of wireless mesh networks sometimes referred to as *client-mesh* where clients also help in routing packets. Mesh technology is becoming increasingly popular for use in wireless network deployments associated with consumer

[103], law enforcement [104], metropolitan [105], and military [106] related use of network services, to name a few.

Infrastructure Wireless Mesh Networks

An infrastructure wireless mesh network [102] is typically a network of WLANs where only one or a few access points are directly connected to the wired world and act as gateways to the public networks. Other access points use multihop routing in order to access one of these gateways or any other access point, effectively forming a wireless backbone. An example of this kind of network architecture is a public network of WLAN hotspots managed by a certain Wireless Internet Service Provider (WISP). For instance, recently, Toronto Hydro Telecom Inc. in Canada turned the whole Toronto downtown into a large WiFi zone [107]. In this network, many access points are deployed throughout a very large area forming a backbone that uses multihop routing to connect mobile users to the wired Internet. In fact, such networks have become quite common around the world. One of the reasons for adopting this type of network architecture is the problem of selfish behavior (avoiding routing packets) or malicious behavior (attacks on the routing protocol) of nodes in mobile ad hoc networks. Therefore, public service providers cannot rely on regular users' devices for packet routing. They may prefer to install a trusted network of routers to guarantee a minimum level of performance and security. Another reason is providing more reliable wireless connectivity than regular MANET networks.

Another example is known as *Wireless Community Networks*, a confederation of WLANs usually meant to provide free Internet access to users. A long list of such

networks can be found in [108]. A realistic scenario of forming a wireless community network is when homes in a neighborhood having access to high speed internet (either directly or indirectly using multihopping and through a connected gateway) allow their home wireless routers to connect with each other and also allow external users in their area (e.g. pedestrians in the neighborhood) to use this network for accessing the Internet.

The company Viasys offers MeshNetworks technology to its customers designed to easily and cost effectively help municipalities, counties and departments of transportation overcome the challenges of rolling out fixed and mobile wireless data networks [109]. Motorola's Mesh Enabled Architecture (MEA) [110] enables vehicles, mobile devices and individuals to connect directly with each other and also to the public telephone network, Internet and private networks for access to voice, video and data services, instantly and securely. It is reasonable to assume that users of the system would not wish for the network operator and all other users to be able to track all their movements simply by taking advantage of weaknesses present in the ad hoc routing protocols. The two main proposals for mesh networking namely SEEMesh and Wi-Mesh have been merged to form a starting point for the 802.11s [92] extension to WiFi standard. The mesh architecture is also being considered by the industry to be used with Wi-Max (IEEE 802.16) technology, instead of the current point-to-multipoint configuration [107, 108].

III.3.2 Privacy Objectives

The design of our protocol depends greatly on our trust model e.g. where the destination's trust lies. In the community network model, we assume that a client trusts

at least one router in its neighborhood (its access router AD) with its location and identifies itself to it in order to be able to receive packets. For example, in a community network consisting of houses, a client may fully trust its access router because s/he is either the owner or a guest in the house. On the other hand, in the WISP model, a client may only trust its home domain while roaming. There may be any number of reasons for lack of trust in other routers including their vulnerability to intrusion, the so-called "big brother" problem and opportunistic public network operators especially when the routes pass through foreign domains. Thus, the design of V-routing for community networks differs from its design for WISP networks, as will be explained.

The location of a client is considered to be known if its access router is known. In the community network model, the locations of the source and the destination must only be known to their own respective access routers. In the WISP model, the access router may not be trust-worthy and thus should not be aware of the identity of its foreign clients. This issue will be explained better later on. We intend to always provide peer privacy to clients. Therefore, the locations of the source and the destination of a connection must be hidden from eavesdroppers (external nodes) and other clients (internal nodes) including each other.

Communication anonymity has to be supported with regards to peers as well as the network. In other words, only the source and the destination must know that they are communicating. We provide this feature by ensuring sender anonymity. Only the destination of a packet can know who the original sender of the packet is. Of course, this has negative implications on reliable packet delivery because intermediate nodes cannot inform the source of any transmission errors and we have to rely on the transport layer

mechanisms such as TCP to implement end-to-end reliable transmission. Besides, link layer techniques for reliable transmission can still be used. Receiver anonymity is provided with regards to all external parties.

III.3.3 V-routing for Community Networks

V-routing protocol consists of three tasks:

1. *path establishment*
2. *data transmission*
3. *path termination.*

Suppose that a source node S wishes to establish a connection to a destination node D. If D is a router, its location in the network is known because it broadcasts routing advertisements. However, if it is a client, our underlying table-driven routing protocols cannot find a path to reach it because it hides its location by refraining from broadcasting routing information. S too, may be a router or a client. If it is a client, the V-routing protocol allows it to hide its location as well. In the following subsections, we first describe weak privacy V-routing for community networks. In section III.3.4, we explain the necessary modifications to this protocol so that it can support the WISP network model.

Part-1: Path Establishment Phase

When an ordinary node D who requires location privacy joins the network, it must acquire at least one of the reachable routers anywhere in the network (preferably a secure and protected router) as its *Rendezvous Point* (RP). This router is used as a transient

destination for any future data packet destined for D. Depending on the device type, network specifications and applications, security considerations and so on the client itself may perform the necessary actions in order to choose its rendezvous point or the client's access router (AD) may perform these tasks on its behalf. In a small non-critical ad hoc network, it is conceivable that a reasonably powerful user device can compute a route towards itself based on the information obtained from the periodic routing update messages broadcasted by routers in the network. However, if the device is not computationally capable or due to security considerations is not permitted to obtain a global view of the network topology then AD may find a rendezvous point and inform the client of the ID of that router. In fact, having the global view of the network topology, AD may calculate a path between RP and itself and convey that information to D.

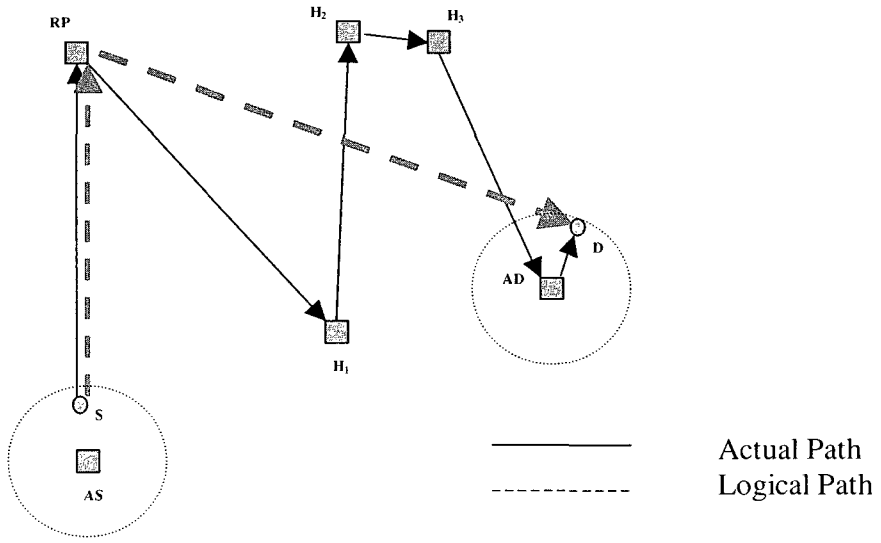


Figure III-3 Triangular path for data transmission in V-routing

Any future source node S trying to send a data packet to D will send that packet to RP first, creating a triangular path as shown in Figure III-3. We denote the first leg of this triangular path with S-RP and the second leg with RP-D. Please note that RP is not by nature a fixed, central or specialized node in any way. Any router in the network has the chance to become at some time the rendezvous point for any client. However, special restrictions such as security considerations may be applied. Also, an arbitrary node in the network may use different rendezvous points at different times. It is possible and more convenient to make the selection of rendezvous points more static and deterministic. Doing so would also reduce the signaling overhead. But this is not recommended because in addition to eliminating the flexibility (in terms of trust and performance) of the client in selecting its rendezvous point, this solution introduces problems of single point of failure, single point of compromise, denial of service, bottleneck, etc. at the rendezvous point.

D (or AD) follows local policies to select RP. Reachability, distance and trust may be some of the criteria. Specifically, one factor that affects this decision is whether D can securely communicate with this router. D must either be able to obtain a signed public key for its RP (if public key infrastructure is used) or it should share a secret key with it (if symmetric cryptography is used). The latter case may be arranged offline by the network administrator before the client joins the network. For example in a public network, a user device may be initially paired with one or more of the base stations distributed across the region. After that, for added security, the two nodes can occasionally negotiate new keys. Also, in systems like infrastructure wireless mesh

networks, implementing PKI is much easier than a generic MANET due to the existence of a fairly reliable infrastructure.

In order for D to be able to establish its second-leg path, we introduce a new unicast message called *forward_req* (forwarding request). D secretly sends several such messages to a few selected routers, which will essentially form a virtual circuit over the RP-D leg. We call these nodes *Virtual Hop Routers (VHR)* because each consecutive pair of them may or may not be physically one hop apart but they are able to reach each other, by using the underlying routing protocol. The format of a forward_req message is shown below:

$$<forward\_req\ ,\ RP\_elect\ ,\ E_{VHR}(2^{nd}\text{-}leg\text{-}id,\ first\_VHR,\ D,\ next\_VHR)>$$

Throughout this thesis, $E_x(M)$ denotes an encryption of message M for node x. If PKI is used, D may obtain the authentic public keys of VHRs from a CA or they may advertise their signed public keys in their routing update messages. Alternatively, if D shares a secret key with the VHR, they can use symmetric cryptography instead of PKI to communicate. This assumption is particularly feasible if all nodes belong to the same administrative domain because the network operator can program each member device with appropriate keys in advance. Of course, a node may share many secret keys with other nodes. Thus, when it receives a packet in which the sender or recipient are encrypted, there must be a mechanism to let the node quickly know which key to use. A

solution may be to let each two nodes assign a globally unique key_id[4] to their shared key that is transmitted between them as plain text within the forward_req message.

A *forward_req* message is delivered by the underlying routing protocol to the recipient VHR in the body of an IP packet (the header of the IP packet contains the address of the VHR but D's address in not specified.). The first field in the packet "*forward_req*" specifies the type of packet. The Boolean value (a TRUE or FALSE value) *RP_elect* indicates to the recipient router whether or not it has been selected as a rendezvous point by D. *2nd-leg-id* is a global identifier chosen by D that uniquely determines this 2nd-leg path and has the same format as IDs[5]. The Boolean field *first_VHR* is used to indicate whether or not the recipient is the first VHR. The need for this field will become clear later. *next_VHR* is the ID of the VHR that follows in the path towards D (i.e. downstream path).

In order to thwart content analysis attacks, the entire IP packet is re-encrypted on each hop. This effect can be achieved using link-layer encryption or IP layer security mechanisms such as IP Encapsulation Security Payload (ESP) [113][6]. Therefore, the packet will look different in a random manner each time it is relayed and cannot be traced. Even if an attacker could trace this packet, the most he could infer is the fact that *some* node located in the area that D happens to be would be going to use RP as its rendezvous point. This does not violate any of our privacy objectives because D's ID and its location are still unlinkable. Even if the adversary knows this path completely, he

[4] Hash functions can be used to generate identifiers that are globally unique. IETF RFC 4122 defines a namespace for globally unique identifiers. Appendix A provides a brief overview of this RFC.

[5] Please see previous footnote.

[6] ESP allows an IP packet to be entirely wrapped and carried as the payload of another IP packet with the option of encryption for the original packet.

would not be able to identify D or data packets destined for it because each VHR that belongs to this path may act as a VHR for many other nodes and also data packets are indistinguishable, as we will explain later.

At the transmission layer in the TCP/IP protocol stack, UDP is used to transmit these packets. TCP cannot be used because its guaranteed delivery service does not work in this case due to our location privacy requirement. The VHR does not know where D is, thus it cannot send messages to it in the reverse direction as is needed for TCP. In order to support reliable transmission of forward_req messages, we have devised an accumulative ACKs method at the V-routing level that will be discussed later in the "Acknowledging forward_req messages" section.

D sends its first *forward_req* message to RP asking it to forward any packets destined for D to a next_VHR, which we denote by H_1. If RP is willing to act as a rendezvous point for D, it broadcasts this decision in the network using an *I_AM_RP* packet. This packet is not encrypted and is readable by every one. Therefore, any node wishing to communicate with D will know it must send its packets to RP first. The I_AM_RP packet is shown below:

<I_AM_RP, RP, D, next_VHR_OK >

This message is broadcasted periodically in order to maintain soft state in the nodes until the rendezvous point is no longer needed at which time D must send an explicit message called *RP_release* to RP to inform it of this decision. This message is described later in Part3 of the protocol which explains the path termination process.

Due to the dynamic nature of mobile networks, there is a chance that the routing tables of D and RP may not match. Therefore, RP may not actually be able to reach H_1 even though it may be willing to act as D's rendezvous point. In the event of that happening, RP sets the Boolean field *next_VHR_OK* in I_AM_RP to FALSE and proceeds to advertising its current routing information. After acquiring the correct routing information regarding RP, D chooses another router as H_1 and sends a new forward_req message to RP. If RP can reach the new H_1, it will set *next_VHR_OK* in I_AM_RP to TRUE. For increased efficiency (less control traffic overhead) and security, RP can delay submitting individual I_AM_RP packets and instead send one such packet for several nodes that have selected it as their rendezvous point (batch acknowledgement). This way, the time correlation between this message and the forward_req message sent by D is obfuscated.

Once RP is established, D sends unicast forward_req messages to H_1 and the other VHRs on the RP-D path. In each of these messages, D specifies the next virtual hop router for the recipient and sets the parameter RP_elect to FALSE. Of course, AD knows that it must forward packets destined for D directly to it on the link layer. D registers 2^{nd}-leg-id with AD using the appropriate link-layer procedures. This way, each VHR knows only its next VHR within the path leading to D. Other nodes and eavesdroppers only know that D is using RP as its rendezvous point. Only D knows the entire RP-D leg. For the purpose of message authentication, D must digitally sign its forward_req messages.

Instead of sending individual forward_req messages, D could broadcast just one long packet containing the IDs of selected VHRs, each one encrypted for the respective recipient. However, we have not chosen this method for several reasons. 1) This

approach would force every node in the network to attempt to decrypt every single ID contained in the packet. 2) The bandwidth consumed by flooding a long packet in the whole network would reduce the scalability of the protocol. 3) This packet would reveal the number of VHRs in use. Secrecy of this value is important to the security performance of the V-routing protocol. 4) One single packet containing all the second leg information reduces the level of difficulty the adversary faces in terms of recovering the IDs of the routers (and consequently the path) compared to when scattered messages are sent. An adversary cannot be sure if s/he has received every forward_req message from D and even if he could, it would still be a difficult task to distinguish them from messages originated at other nodes. Figure III-4 depicts the path establishment phase in an example network.

Another alternative to individual forward_req messages is using onion cryptography. D could create an onion that would be relayed from one VHR to another until it reached RP. In each layer of the onion, D would include the previous and the next hop for the intended recipient VHR. However, this method incurs more transmission and computational overhead. An onion must be padded after each retransmission in order to always maintain the same length, so that content analysis attacks can be prevented. Therefore, if there are n VHRs and one RP, a forward-req onion would be at least as long as n IDs throughout the path from D to RP. Let's assume that the distance between D and H_i (the i^{th} VHR, where RP is H_0) is h_i for $i \in [0,n]$. Thus, the total communication overhead of onion cryptography would be $(n+1)h_0$ worth of IDs. On the other hand, our individual messages carry only one ID each. Thus, the total communication overhead of

our method is $\sum_{i=0}^{n} h_i < (n+1)h_0$ because $\boldsymbol{h_i < h_0}$ if i is not zero. Our savings are actually more than this because here we have assumed that the individual messages travel along the same path that the onion would take. However, in reality, the underlying routing protocol which is responsible for delivering these packets may choose a shorter route (shortest path) for each VHR.

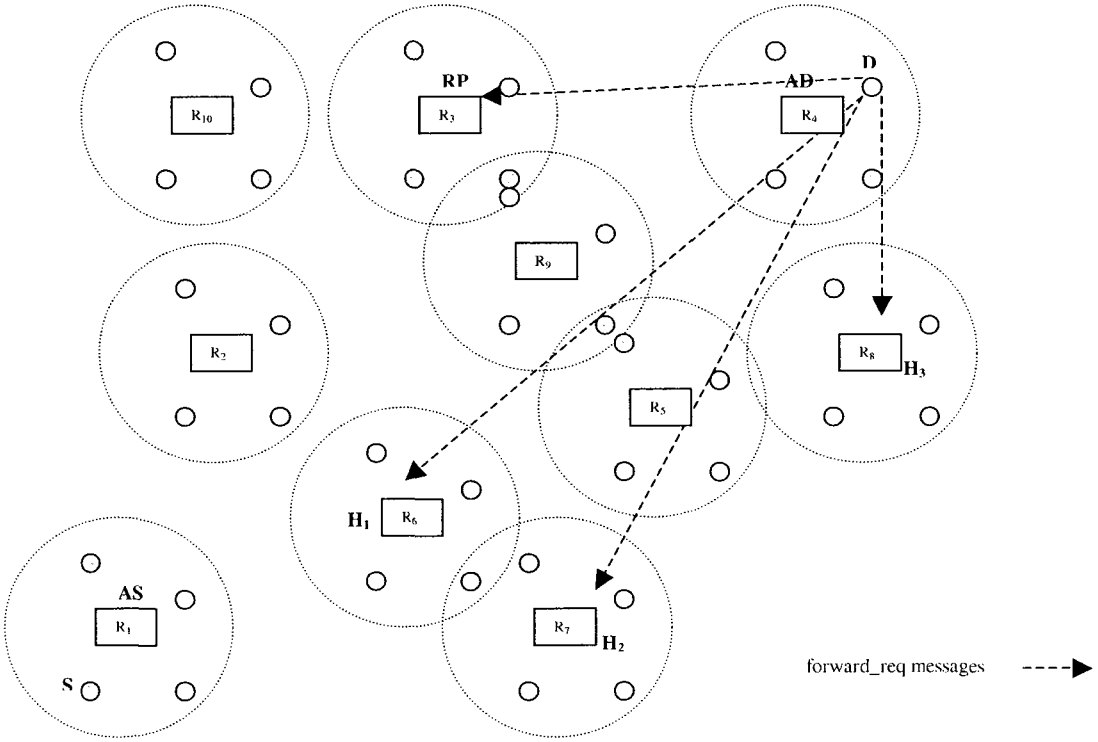


Figure III-4 Path establishment phase – forwarding requests

In terms of computational overhead, our method is more efficient too because we require each VHR to decrypt only one encrypted ID while onion cryptography requires them to perform multiple decryptions.

If RP is not willing to act as the rendezvous point for D, it will refrain from broadcasting an I_AM_RP packet. Therefore, D can interpret the receipt of such a packet as an ACK from RP and its absence as RP's unwillingness (a NACK), in which case it may try RP again (in case the previous message was lost.) or choose another router and repeat the same process. Making RP, instead of D itself, responsible for disseminating this information (the role of RP as D's rendezvous point) in the network has three benefits. 1) It prevents a nearby eavesdropper from locating D. Note that D's ID is not encrypted in I_AM_RP packet because we want every node (potential source nodes for D in future) to become aware that D is using RP as its rendezvous point. 2) It reduces the overhead of the protocol because it lets RP to implicitly send an ACK to D (piggybacking). 3) RP is able to take advantage of advanced flooding techniques already in use by the underlying routing protocol such as the "multipoint relay" technique of OLSR to further reduce the overhead.

D (or AD) needs to ensure that its 2^{nd}-leg path(s) is always connected. Therefore, it must receive a notification from the underlying routing protocol when a change in the network topology is detected. D may select multiple RP's and establish more than one path towards itself. This action promotes load balancing and multipath routing.

Reliable Delivery of forward_req Messages

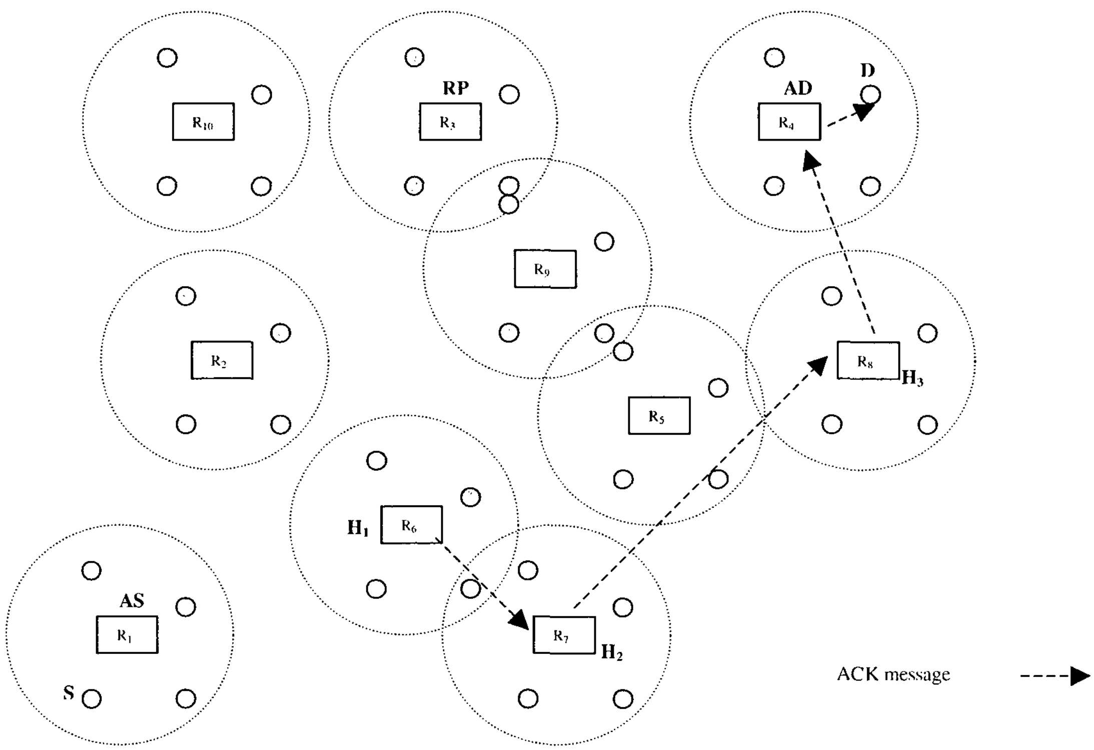
As was explained, RP acknowledges a forward_req message with an I_AM_RP packet. Other VHRs use a different mechanism for this purpose that carries a cumulative ACK message from all VHRs back to D. To enable this mechanism, we use the Boolean field *first_VHR* in the forward_req message, which is only set to TRUE for H_1. After

receiving a forward_req message, H_1 assembles an ACK packet, which contains a long fixed-length randomly filled data structure called *magazine*. We have chosen this name for this field because of its similarity to the magazine of a machine gun where the bullets are pushed in on top of each other and are later released in the reverse order. Another analogy for this field is a First-In-Last-Out memory stack. The format of this ACK message is shown below:

$$<ACK,\ E_{next\_VHR}\ (2^{nd}\text{-}leg\text{-}id),\ magazine>$$

H_1 pushes its ID (encrypted for D) at one end of the magazine and sends the message to H_2. H_2 and every other VHR on the path down to and including AD do the same. If VHRs use D's public key, each of them must first append a nonce (a one-time random number) to its own ID. Otherwise, a compromised intermediate router could systematically try each ID in the network encrypted with all the available public keys to eventually uncover each address in the magazine. A compromised VHR would be in a better position to mount this attack because it knows it only needs to try the public key of D. When a nonce is used, only D, using its private key, can uncover each filed in the magazine. Delivering an ACK packet from one VHR to the next one is the responsibility of the underlying routing protocol. Each VHR determines its next VHR based on the 2^{nd}-leg-id. Every VHR decrypts this field and then re-encrypts it with the public key of its next VHR. Alternatively, each two consecutive VHRs can use symmetric cryptography. For this to be possible, D can distribute pairwise shared keys to each two adjacent VHRs in the body of the forward_req message. Instead, each VHR may initiate a Diffie-

Hellman key exchange process with its downstream VHR. These keys can then be also used in the data transmission phase. Per-hop re-encryption at the link layer is applied to the whole packet in order to prevent content analysis resulting in uncovering D's location.



$\{ <E_D(AD)> <E_D(H_3, n_3)><E_D(H_2, n_2)><E_D(H_1, n_1)> \ldots\ldots\ldots\ldots\ldots\ldots \}$

← random bits→

final magazine delivered to D

Figure III-5 forward_req acknowledgment process

If the ACK message traverses every VHR and successfully arrives at AD and is then forwarded to D, D will know that the second leg is complete. However, if some H_i is

unable to reach H_{i+1} it will broadcast the so far accumulated ACK message. H_i encrypts the 2nd-leg-id for D. On the other hand, after waiting for a specified amount of time, H_{i+1} issues an ACK message if it does not receive one from a previous VHR. At the end, D may end up with a few contiguous segments with gaps between each two. At that point, for every *hanging* H_i (a VHR unable to reach its next VHR) D selects a new H_{i+1} (reachable by H_i and able to reach the old H_{i+1}) and sends a new forward_req message to H_i. D also sends a forward_req message to the new H_{i+1} specifying the old H_{i+1} as its next virtual hop router. By setting the *fisrt_VHR* to TRUE, D instructs the first hanging VHR in the chain to originate an ACK message and the process continues like the first time.

We could let RP initiate the acknowledgment process instead of H_1 in which case there would be no need for the *first_VHR* field in the forward_req message. However, that would require an unnecessary transmission from RP to H_1, which would consume time and resources of nodes and the network. Figure III-5 shows the acknowledgement process for the example network of Figure III-4 as well as the final magazine where "n_i" represents a nonce. AD does not need to append a nonce to its ID since it knows that the next recipient of the magazine is D itself.

An alternative method to using the magazine field that is perhaps more bandwidth-efficient and also allows D's ID to be omitted from the forward_req messages (for better receiver anonymity) is the following. Instead of the magazine field, H_1 encrypts a well-known string with its private key (or with its key shared with D if we are using symmetric cryptography) and forwards it to the next-hop VHR, which in turn encrypts that field with its private key. Every VHR on the 2nd-leg would add one layer of

encryption to this field until it is received by D. D knows the sequence of VHRs that it has chosen for this specific 2nd-leg and can apply recursive decryption using the public keys or shared secret keys corresponding to these routers. Once it gets the well-known string, it knows that it has to stop decrypting. Please note that the layered cryptography does not change the packet length over consecutive virtual hops.

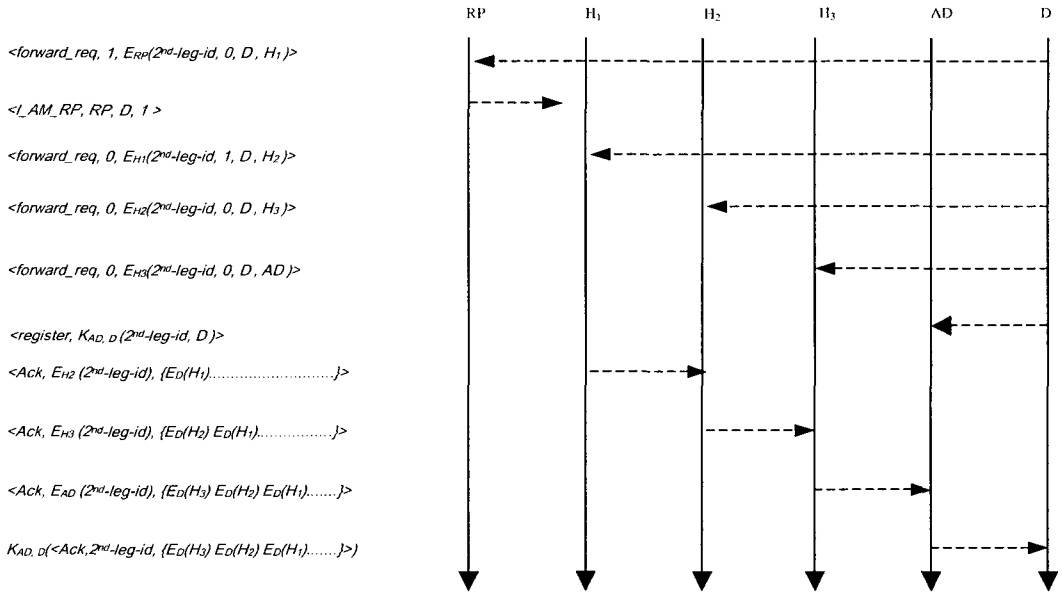


Figure III-6 Time diagram of V-routing path setup phase

Figure III-6 shows a time diagram and the exchanged messages in a path establishment process involving 3 VHRs.

Part-2: Data Transmission Phase

When a source node S wishes to send a packet to D, it readily knows which router it can use as a rendezvous point. Due to the proactive nature of our protocol, no route acquisition is necessary and therefore no interaction between the source and the

destination is needed. This feature helps communication anonymity because an attacker cannot correlate the route request messages from the source and the route reply messages from the destination, as is the case in reactive routing. All **data** packets in the network, at the routing layer, are one of two types; *direct* or *indirect,* both having the format shown below:

$$<dest, src, final\_dest, payload>$$

In a direct packet, '*final_dest*' which is the ultimate recipient is the same as '*dest*'. In indirect packets however, '*dest*' is the address of a transit destination while '*final_dest*' is the address of the ultimate destination. The underlying routing protocol is only concerned with dest while the V-routing layer examines final_dest as well. In other words, as far as the underlying routing protocol is concerned, final_dest is part of the payload. In fact, the V-routing sublayer at the source and each VHR determines and appends dest to the data packet based on final_dest and then passes that to the routing sublayer.

When a node receives an indirect packet it forwards the contents of it to the final destination "*final_dest*". Thus, S sends its data packets as indirect packets to RP as *dest* with D as the *final_dest,* as shown below.

$$<RP, E_D(S, nonce), E_{RP}(D, nonce), E_D(payload)>$$

The main purpose of nonce is similar to what was explained in the path establishment phase but it has another benefit as well. If nonce is not used, even if the adversary is not able to uncover the real IDs, it can gain valuable knowledge about the traffic patterns in the network based on fixed encrypted IDs. In a way, nonces create frequently changing pseudonyms for ordinary nodes.

Per-hop encryption on the first leg is not necessary because even though a global eavesdropper can follow the packet to RP, the only thing that it can conclude is the fact that some node in the vicinity of AS is sending a packet to RP. RP understands that the payload of the packet must be forwarded to D, the final destination. S is encrypted for the destination thus the communication is anonymous to all other nodes even the two access routers AS and AD. However, this action precludes the possibility of informing S by intermediate routers if its packets are, for some reason, not delivered to D. Therefore, S should work on a timeout basis and we can only rely on end-to-end mechanisms such as TCP for reliable transmission. If an acknowledgment does not arrive from D in a timely manner, S will retransmit the packet. Knowing S, D can determine a rendezvous point for it and send acknowledgements to it as regular data packets with S as the final destination. In other words, D must send acknowledgments to S in the opposite direction, using a path established by S.

When RP receives an indirect data packet destined for D, it replaces D's ID in the packet with the appropriate 2^{nd}-leg-id encrypted for H_1 and forwards the packet to it. This field is re-encrypted at each VHR. In order to prevent content analysis using the un-mutating fields of source ID and payload, they are also re-encrypted at every VHR.

Therefore, per-hop encryption of data packets on the second leg is not necessary either. A data packet forwarded from H_{i-1} ($i > 0$ to include RP as well) to H_i looks like:

$$< H_i, E_{Hi}(E_D(S, nonce)), E_{Hi}(2^{nd}\text{-}leg\text{-}id, proof), E_{Hi}(E_D(payload)) >$$

The well-known string *proof* has the same length as nonce and has two purposes. First, it makes the packet the same length as the packets on the first leg. In fact, all the data packets in the network must be of the same length to prevent content analysis. The source may apply necessary padding to the payload before transmission. The second purpose of proof is to let a VHR verify that it is indeed the intended recipient of this packet, if symmetric cryptography is used between it and the previous VHR. Every VHR determines its next VHR based on the 2^{nd}-leg-id and modifies the packet accordingly. At the last hop, the payload and $E_D(S, nonce)$ are forwarded by AD to D on the link layer using their shared secret key. These indirect packets are transported using regular IP protocols, e.g. according to the IP encapsulation specifications of the IETF RFC 2003 [113]. The actual and the virtual paths established between the source and the destination are shown in Figure III-7.

Part-3: Path Termination Phase

When the RP-D lag is no longer needed by D, it must be torn down so that nodes' storage spaces are released as well as potential future packets destined for D are not routed incorrectly. For this purpose, D uses a special unicast V-routing message called *RP_release* . The format of this message is as follows:

$<RP\_release, E_{RP}(D)>$

This packet is delivered to RP using the underlying routing protocol similar to a forward_req message. The field *RP_release* indicates the type of packet. Once RP receives such a packet, it stops broadcasting periodic I_AM_RP packets corresponding to node D. Each VHR involved in the RP-D leg also receives such a message from D and removes its corresponding soft state. Alternatively, they could learn of this fact due to absence of periodic I_AM_RP packets from RP.

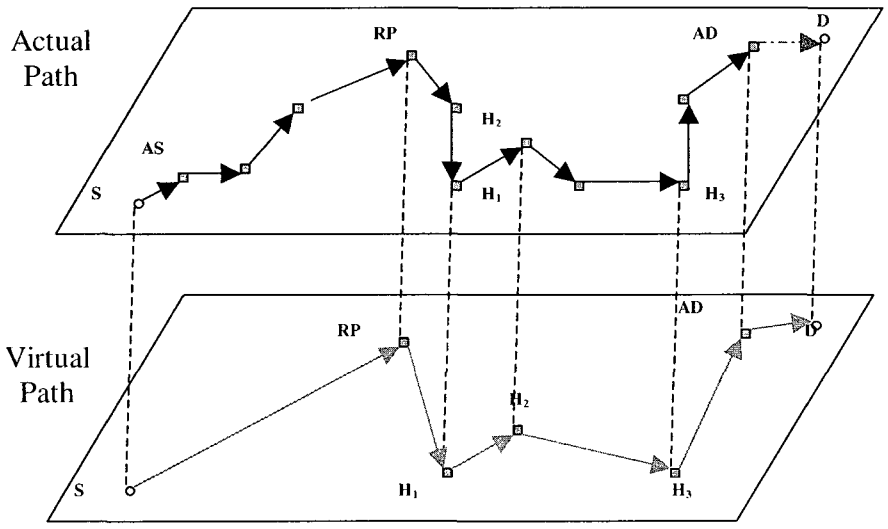


Figure III-7 V-routing data transmission phase

III.3.4 V-routing for WISP Networks

A WISP network is similar to Mobile IP (MIP) [101] where a *Home Router* would play the role of the home agent and the access router would be considered a foreign agent. In Mobile IP, while away from its home domain, a mobile agent (client) registers a Care of

Address (CoA) that it obtains from a foreign agent with its home agent so that it would remain always reachable. A packet intended for the mobile agent would be delivered to it via its home router who would forward it to the registered CoA. In order to improve efficiency, a route optimization technique is usually used in Mobile IP where the home agent informs an initiator of the CoA of the mobile agent (recipient) so that they can communicate directly. This packet delivery scheme causes a security problem because a home agent is always aware of the current location of its mobile agents and can track their movements. In Mobile IP with route optimization, this problem is even more severe because a source node would also be informed of the location of the destination node. In the context of mobile networks, this problem has been addressed in [114] using the non-disclosure method. However, that method is not suitable for an ad hoc network.

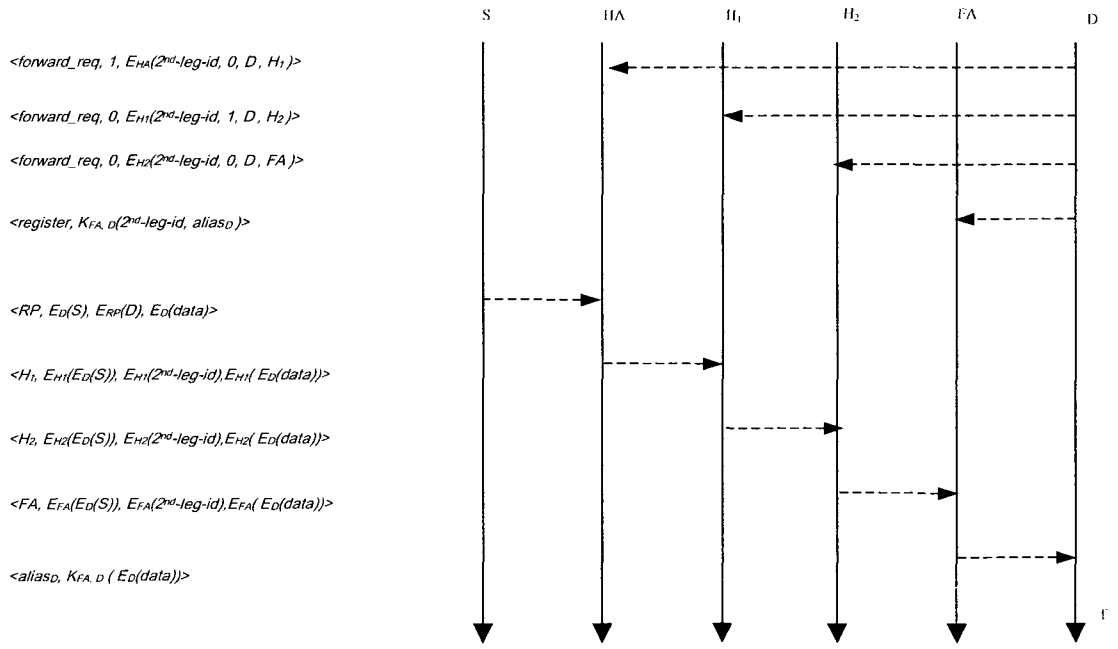


Figure III-8 V-routing for WISP

V-routing can be used for this network model with a slight modification. A client must register with its access router with an alias instead of its real network ID. An anonymous authentication protocol can be used between these two nodes if needed. The home router would act as the rendezvous point. In fact, it would be a well known fact that RP is D's home router and there is no need for an I_AM_RP message. However, D still registers a 2nd-leg-id and a next_VHR with its home router. The second leg would still be configured by the client based on a 2nd-leg-id and the rest of the protocol would be unchanged. This way, the home router would be unaware of the client's location and the access router would be unaware of its real identity or the identity of its home router. The access router distinguishes packets destined for D based on the 2nd-leg-id. Figure III-8 shows the path setup and data transmission phases in this case but the acknowledgment messages have been omitted.

III.4 V-ROUTING WITH STRONG PRIVACY

This version of V-routing supports a network model in which the client does not have any trust relationship with other entities and therefore it does not share any information with other nodes from which its location can be inferred. This protocol is based on using aliases in addition to real network identifiers where the mapping between the real ID and the alias will only be known to the client itself. In other words, a client invents an alias for itself in the same format as addresses used for routing. It then advertises this alias in the network via routing updates. All nodes participate in routing but the nodes that require location privacy introduce themselves with their aliases in routing updates. The real ID of the client will be used by a potential source node to refer to it and send packets

to it but its alias is used for the purpose of routing packets. In other words, each node has a global view of the network topology in terms of aliases but cannot relate any of aliases to its corresponding real ID except its own. For the purpose of secure routing, the alias may be authenticated independently of the real ID. The 2^{nd}-leg is established based on the alias and a 2^{nd}-leg-id as in the weak-privacy version of V-routing. However, instead of the ID of the access router, the forward_req message sent to the last VHR contains the alias of the client.

The price of strong privacy in this protocol is the extra overhead of routing updates and the fact that all nodes must participate in the routing protocol, i.e. all nodes have to act as routers. Otherwise, a node that declares its unwillingness to route packets would be revealing itself as the destination for a specific 2^{nd}-leg while establishing it. Because, the destination (D) sends a forward-req message to the last VHR in the path instructing it to forward packets to an alias which happens to be D's. Thus, if D has already announced that it would not act as a router, at least that VHR finds out that D is actually the destination for that 2^{nd}-leg. The overhead of routing update in this version is higher because every node, not just the access routers, has to generate them.

This network model matches MANET and mesh-client networks very closely because all nodes have the same roles and security requirements. An example of this type of network could be a mobile ad hoc network formed by military forces, police or other law enforcement personnel, vehicles and offices. In this case, a compromised node could identify and locate important nodes (e.g. command posts or top-secret weapons) that communicate with it as the sources or destinations of data flows. Also, an eavesdropper

could trace packets to their sources and destinations. V-routing can prevent these security risks in this kind of network.

The transmission phase of this protocol is depicted in Figure III-9, where S is sending a packet to D. The entities in braces indicate the information that a VHR has about this connection. Each VHR only knows the alias of the next VHR on this 2^{nd}-leg and as far as H_3 knows $alias_D$ is just the alias of another VHR.

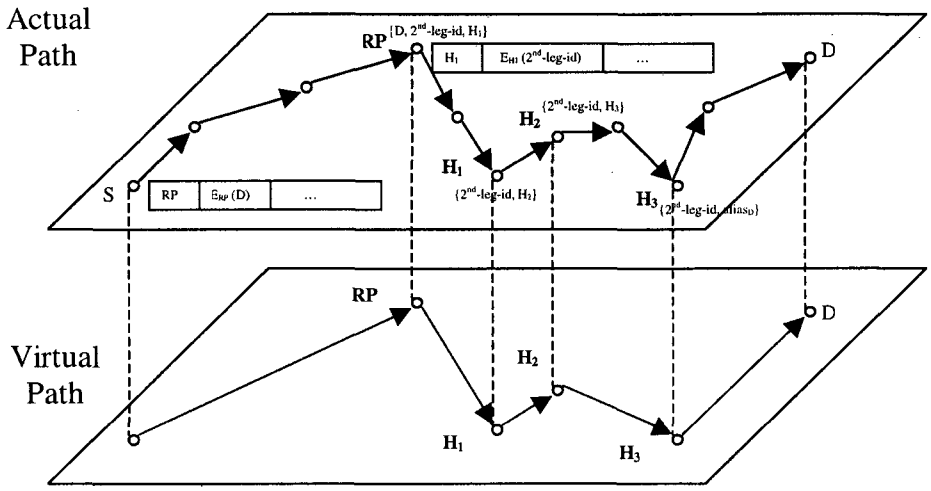


Figure III-9 Strong privacy V-routing data transmission phase

III.5 SECURITY ANALYSIS

When we began, we set out to provide 3 main security services: 1) communication anonymity 2) source location privacy 3) destination location privacy.

Communication Anonymity

This objective is achieved mainly owing to the proactive nature of the protocol. The second leg of the path is established by the destination regardless of the source and before a connection is needed. Especially, unlike on-demand routing protocols, route request and route reply messages cannot be correlated. During data transmission, the source identity is always encrypted at least for the destination, which ensures communication anonymity as well as source anonymity as an added benefit. Destination ID is encrypted for the rendezvous point on the first leg and is not used on the second leg, which again helps communication anonymity. Destination anonymity is always preserved with regards to eavesdroppers. Finally, by making data packets indistinguishable, we have made packet tracing during data transmission impossible. There is only one situation where total communication anonymity may be violated namely in weak privacy V-routing for community networks if the source node does not have enough storage capability to maintain RP's for all other clients. In that case, it may request this information from its access router who supposedly can keep this information. Therefore, the access router of the source would know that S is going to communicate with D. But, this is not a big concern because we assumed that access routers in community networks are trust-worthy. In the WISP model, the access router does not know the real identity of the source. In the MANET network model, if we assume that nodes are not computationally powerful, then we have to assume that the mappings between nodes and their rendezvous points are kept by a directory service in which case the source can safely retrieve its needed information from this server using its own alias.

Source Location Privacy

The main reason behind location privacy for the mobile nodes is the fact that they either do not send routing updates (weak-privacy V-routing) or decouple their real identities from the routing protocol (strong-privacy V-routing). The only case where total location privacy is violated is in community networks where the access router of a client knows its ID and its location at the same time. In the WISP model, the access router does not know the real ID of the client and the home router does not know its location. Source location privacy cannot be violated during path establishment because it is not involved. During data transmission, the source location privacy stems from source anonymity.

Destination Location Privacy

Preserving destination location privacy has been our most important objective. The rendezvous point and the VHRs are aware of the identity of the destination but they only know their next VHR to reach D not the whole path. Even if they all collude together, they will not be able to locate D unless the access router of D colludes with them. This is similar to the mix nets where having only one honest mix node is enough to preserve the anonymity of the connection. Strong-privacy V-routing is more secure because the last VHR only knows the alias of D. Each forward_req message is encrypted at the link layer or IP layer and cannot be traced back to D. Sending these messages to individual VHRs separately is another factor in making the protocol more secure. In protocols based on mix networks or onion routing, the identities of all the intermediary nodes are contained in the same path setup packet. If an adversary intercepts such a packet and happens to know the private keys of all those nodes, it is able to reveal the whole path. In V-routing,

even if the eavesdropper knows all the private keys, it still needs to intercept all the forward_req messages, which is less likely. Even in that case, unless it knows exactly how many VHRs are being used, it cannot find the location of the destination because it does not know for a fact that it has captured the forward_req message sent to the last VHR.

If the attacker can be sure that it receives all these messages and that it can decode them, then it can easily reconstruct the complete second leg and figure out which router is AD. Theoretically, this is impossible because D is the only entity that knows how many messages it has submitted but if the attacker has a complete coverage of the network it can be reasonably sure that it will receive all wireless packets. Another way to achieve this effect is to assume that all the virtual hop routers collude together and share their information about the second leg. Presence of AD in this collusion is crucial because if it does not participate the attacker can only infer that the home router of D is one of the remaining routers in the network, which is not very helpful if there are many routers. In fact, if the attacker has the cooperation of AD, it will not need help from any other router to find this particular destination node.

This is indeed a very strong attacker model. It requires the assumption that an attacker is capable of receiving every single wireless packet in the network, which is very improbable in a wireless network. Then, we have to assume that it has access to all the secret keys or can break the cryptographic system. Today's encryption systems are very powerful and this event has a very low probability of happening. Alternatively, we have to assume that all the virtual hop routers are malicious. In practice, an attacker may be able to gather partial information about the second leg. Every successfully captured and

decoded forward_req message eliminates the recipient VHR as a possibility for AD. Also, according to the protocol, RP is known to everyone. Suppose that there are N routers in the network aside from RP and the second leg consists of n virtual hop routers. Now, let's assume that the attacker knows all the links on the second leg. Yet, it cannot be sure of that fact. Therefore, it can only eliminate n VHRs as probable access routers for D.

We found that an attacker can at the best guess that D belongs to one of N- n routers but for that it has to be able to receive and decode n forward_req messages. Contrary to the initial intuition that a larger number of VHRs increases the location privacy, we can see that it actually reduces it. However, this is because we have assumed that the attacker is in the possession of all the forward_req messages. But, if p is the probability of the attacker receiving a message on average and q is the probability of it being able to decode that message then the probability of acquiring all the necessary information by the attacker is $(pq)^n$. The larger the number of VHRs the smaller this probability is. This probability can be used as a measure of strength for our location privacy scheme. A large n indicates a high level of privacy but the user may have to pay the cost of using network resources. The choice of n must not be specified by the protocol and must be up to the user in order to minimize attacker's basic knowledge of the protocol operations.

III.6 PERFORMANCE ISSUES

III.6.1 Protocol Overhead

Compared to other routing protocols, our protocol does not seem to have excessive overhead. In the path establishment phase, we use very short forward_req messages (almost 3 IDs worth long each) and ACK messages that produce an overhead less than or comparable to the route acquisition phase in reactive routing protocols. Specially, considering that in reactive routing protocols, every source node has to perform the route acquisition while in V-routing only the destination establishes the 2^{nd}-leg path once for all its future connections. This overhead depends on the number of VHRs that are used. The overhead of I_AM_RP packets is usually less than the savings gained by the elimination of routing updates on behalf of ordinary nodes in the mesh network model. In the MANET network model, this overhead has to be considered but it is also small (2 IDs worth per node and flooded therefore $2N^2$).

In the data transmission phase, some overhead is produced due to the padding added to IP packets at the source node, which is not significant if the application layer generates a reasonable amount of data. The other source of overhead in this phase is the end-to-end triangular path. This overhead and its associated latency are comparable to protocols such as Mobile IP [101] and mix nets.

III.6.2 Computational Overhead

The asymmetric cryptography overhead of the path establishment phase of V-routing is comparable with most secure routing protocols. Once the VHRs are determined, they can

exchange symmetric keys to be used in the acknowledgment and the data transmission phases, using for example Diffie-Hellman key exchange protocol. The cryptographic overhead of our data transmission phase is caused by IP tunneling (the packet is encrypted before adding the new header) similar to virtual private networks.

In terms of state complexity, the soft state maintained at every VHR is a small amount of data including a 2^{nd}-leg-id and a next VHR. This is similar to a regular proactive routing protocol.

III.7 KEY MANAGEMENT SYSTEM

Because of the scalability issues of the symmetric cryptography key management systems (due to key sharing requirement), at least in its path establishment phase, V-routing uses asymmetric cryptography. Therefore, it relies on a Public Key Infrastructure (PKI). Recently, there has been a great deal of effort towards making PKI viable for ad hoc networks. One of the most promising techniques in this area is threshold cryptography. As far as ad hoc networks are concerned, a particularly challenging aspect of PKI is its need for a central trusted entity usually referred to as a Certificate Authority (CA) that can provide authentic public keys of an entity to a requesting third party. Nevertheless, all secure routing protocols for ad hoc networks rely on a central trusted authority or must have pre-shared keys.

III.7.1 PKI for Ad hoc Networks

Among the many proposals for managing trust in multihop ad hoc networks, the distributed PKI scheme seems the most promising. Using Shamir's groundbreaking idea of threshold cryptography [115- 117], Zhou and Haas proposed a key management system based on certificates [53] for ad hoc networks. By distributing trust among multiple specialized servers instead of just one, they implemented a virtual certificate authority. In a (k, n) threshold scheme a correct signature can be produced as long as k out of n servers are correct (for example present or un-compromised). These n servers constitute one distributed CA that as a whole has a public key pk_{CA} known to every node but its private key sk_{CA} is divided into n parts (secret shares), one for each server and no single server knows the complete secret private key sk_{CA} used to sign certificates. Each server can use its secret share to produce a partial certificate for a requesting node called a *client*. These partial certificates can be combined by one of the servers called a *combiner* to produce the complete certificate. A certificate cannot be obtained unless at least k correct partial certificates are available.

Considering the amount of viable proposals in the literature, we believe that the availability of some sort of distributed PKI for mutihop ad hoc networks can be assumed. Therefore, in the next section, we describe our public key retrieval system for V-routing based on availability of a certificate authority.

III.7.2 Certificate Management in V-routing

In V-routing, a node needs to anonymously retrieve the authentic public keys of other nodes from a CA in a way that its own identity and location are not compromised. Below, we will show how V-routing itself can be used to achieve this objective.

Similar to existing protocols, we assume that routers broadcast their public key certificates along with their routing updates. They obtain these certificates directly from the CA because they do not hide their locations. However, a client D obtains and distributes its certificate in an indirect fashion as explained here. Having the authentic public keys of its chosen RP and VHRs, it establishes a 2^{nd}-leg for itself. Then, it sends a certificate registration message (cert_reg) to the CA who sends this certificate to RP within a certificate issuance message (cert_iss) in order to be included in its next I_AM_RP message for D. In strong privacy V-routing, all nodes are both routers and clients and each must certify a public key for its client ID.

In the data transmission phase, a source node S may obtain the certified public key of the destination node D from its own access router AS. However, if access routers do not maintain certificates of ordinary nodes but only those of other routers, S must retrieve this information directly from the CA. In order to do that, S would establish a 2^{nd}-leg for itself as the destination node in its connection with the CA. Then, it would send a request for D's certificate (cert_req) to CA which would return this certificate via S's rendezvous point. In fact, this way, the communication anonymity is preserved with regards to AS because CA encrypts D's ID and its certificate with S's public key, which it knows because it has already signed it. The timing diagram of Figure III-10,

demonstrates these message exchanges for weak-privacy V-routing. The full format of the cert_reg message issued by a node X is as follows:

$$\text{cert\_reg } (X, sk_X, pk_X, RP_X, \text{init\_cerd}_X)$$

sk and pk represent the private and public keys of node X and init_cred represents its initial security credentials that proves its identity to CA. A router does not need to specify a RP in its cert_reg messages.

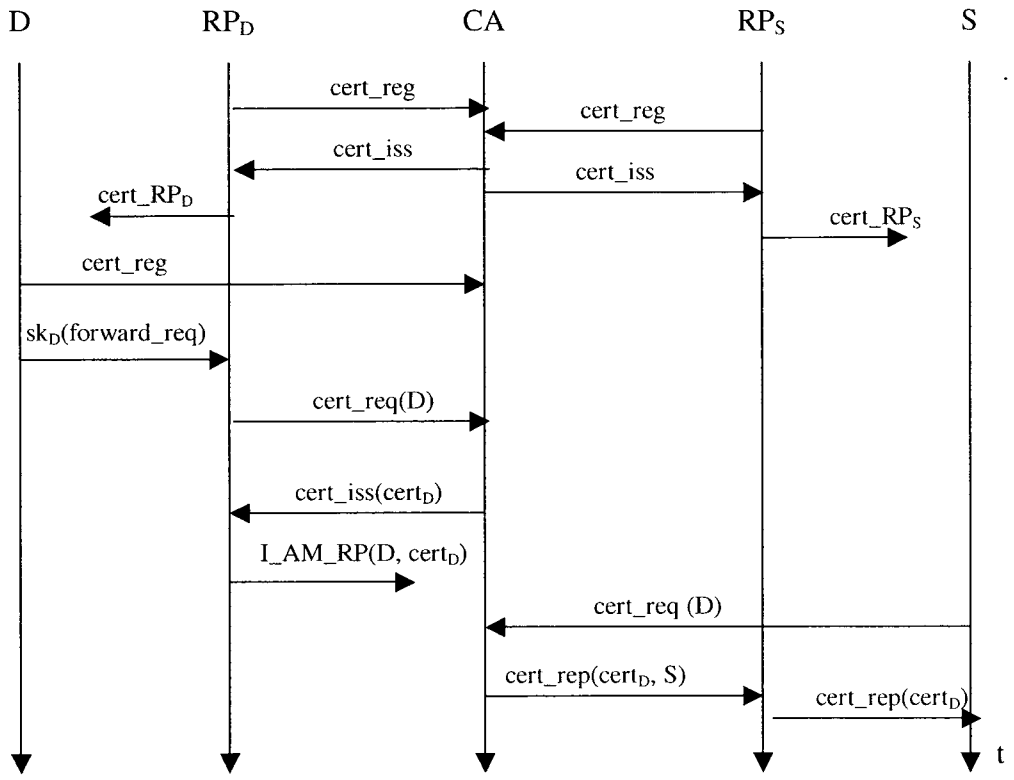


Figure III-10 Certificate registration and retrieval in V-routing

III.8 V-routing for Anonymous Multicasting Applications

Multicasting is used in a network mainly in order to minimize the amount of duplicate packets and hence save bandwidth when a source node sends the same data flow to multiple destination nodes. The routes between the source node and its multiple destinations may partially overlap. Therefore, it is possible to transmit only one copy of each packet on the shared part of the path and then duplicate it for each destination at the end of the shared path. For example in Figure III-11, the routes S-D_1 and S-D_2 overlap on the S-R_1-R_2 segment. In this case, S can transmit each packet only once for both destinations, which will be relayed by R_1 only once. However, R_2 sends a copy of the packet to D_1 and another to D_2. To achieve this objective, a multicast tree must be established where S is called the *root* of the tree and each destination is called a *leaf* or a *subscriber*. The set of all subscribers is called the *multicast group*.

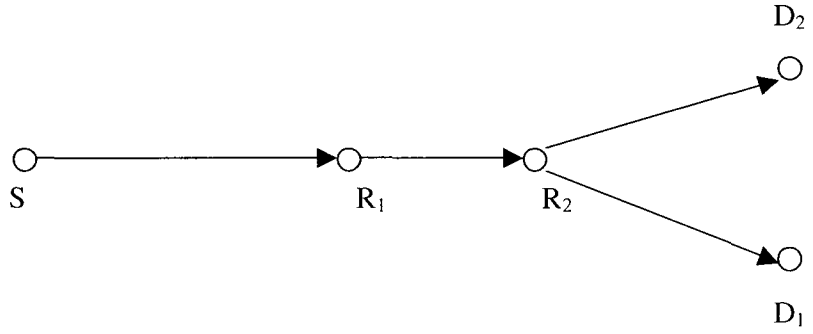


Figure III-11 A simple multicast tree

With respect to the multicast group, two types of multicasting can be envisioned:

Addressed Multicast: The source node determines the destination nodes. Examples of this type of multicast are bulk email, videoconferencing, distance learning and distribution of software or specific data to clients of a company.

Open Multicast: In this type of multicast, normally anyone is allowed to join the multicast group although sometimes authorization may be needed. An example of this kind is providing online content such as news, stock quotes and entertainment. In these applications, the identities and the locations of the destination nodes are not significant. For example, in IP multicast protocols all interested recipients subscribe to an advertised multicast IP address in order to receive data from a certain multicast session distributed to that address. Usually, the data is sent from the source to multicast routers across the network, which then duplicate the packets for downstream subscribers. A subscriber usually joins a nearby local multicast router. Standard anonymous membership and authorization methods may be used to hide the identity of the recipient. Some work on the anonymous multicast group subscription has been done e.g. [118]. Another important aspect of multicast communications namely group key management has been the subject of many papers. In this section, we adapt V-routing for addressed multicasting applications.

III.8.1 Multicast V-routing

Addressed multicast can be one of two kinds:

1. *One-pass multicast*: The source selects its destination nodes and immediately sends its data to them e.g. in bulk email applications. There is little that can be done in this case in terms of efficiency because there is virtually no opportunity for establishing a suitable multicast tree. In terms of anonymous routing, the source sends a copy of its packets to the pre-established rendezvous point of each destination node. If two or more destination nodes share the same rendezvous point, the source can send only

one copy of the packet using a procedure similar to what will be explained in the invitation-based multicast next.

2. *Invitation-based multicast:* In this case, the source advertises a multicast session by sending invitations to destinations of interest to join its multicast group. In order to have a more efficient multicast tree, the source specifies a common rendezvous point that will be the root of the multicast tree. Indeed, this design coincides with the shared-tree method of the Cisco IP multicast protocol where all of the multicast group members may act as a source or a destination node. If there is only one source and it does not need location privacy, it may become the root. Also, if a destination node is not concerned with location privacy, it can simply inform the root of the tree of its current access router (in weak-privacy V-routing) or its alias (in strong-privacy V-routing) and let the root compute the best route to that destination. Otherwise (destination needs location privacy), the destination will setup a hop-by-hop path downward from the rendezvous point similar to regular V-routing. If the multicast group is a mixture of these two types of subscribers, the root can setup a multicast tree for the set of subscribers that do not require location privacy, using the underlying routing protocol, provided it has support for multicasting. For the set of subscribers that need location privacy, the source knows only the next hop router for each one of them. If any of these routers is the same for two or more subscribers, some saving is there to be gained by transmitting only one copy of each packet to it.

In the next section, we will describe our invitation-based anonymous multicast protocol based on V-routing. Similar to V-routing, this protocol preserves location privacy for the source and all the destinations. Also, it preserves communication anonymity with regards to all third parties so that only the source and each individual subscriber are aware of their connection. Moreover, for each two subscribers, no third party other than the source, not even the subscribers themselves know that they are subscribing to the same multicast group. Source anonymity and destination anonymity are also preserved.

Invitation-based Multicast

Our multicast protocol makes use of 6 types of control messages:

1. *Session Initialization (sess_init):* The source sends this message to the rendezvous point informing it of its intention to use it as the root of its multicast tree.

   $$\{\text{sess\_init, RP, } E_{RP}(g, tk)\}$$

   g is the multicast group address identifying the multicast session and tk is a token that allows RP determine who is allowed to subscribe to this session. S sends this token to each intended recipient using a separate session advertisement message described next.

2. *Session Advertisement (sess_adv):* The source sends this message as a data packet to each destination D using their individual rendezvous points inviting them to subscribe to *g* with RP as the root of the tree and using the token *tk*.

   $$\{\text{sess\_adv, g, RP, tk}\}$$

3. *Join Request (join_req):* After receiving an invitation, each interested destination node uses this message to establish a second leg for itself starting at RP. This is similar to the regular V-routing and ensures location privacy and anonymity of the destination. However, allowing each destination to establish a random and independent 2^{nd}-leg would result in an inefficient multicast tree due to little path sharing. Instead, we use a 2-pass approach using onion routing for this phase. Because the location of RP as the destination of this message is not a secret, each destination can generate an onion corresponding to the shortest path between itself and the group rendezvous point. Each layer of this onion is encrypted for the upstream router towards the RP and contains *g* and the 2^{nd}-leg-id. The innermost layer is encrypted for RP and contains tk as well which proves to RP that this particular 2^{nd}-leg belongs to an authorized recipient. The format of this message is shown below:

$$\{\text{join\_req}, E_{R1}(l, g, E_{R2}(l, g, \ldots E_{RP}(l, g, tk)))\}$$

As in standard onion routing, each router on the way memorizes from which neighboring router it receives this message and adds a record to its Multicast Information Table (MIB) containing that router, l and g. This router would be one of the downstream next-hops for the local router during data transmission to the multicast group g. Each router rebroadcasts the onion after peeling off its own layer until it reaches RP. An example MIB is shown below:

Table III-1 An example Multicast Information Base

Group	2^{nd}-leg-id	Next hop
g_1	l_1	R_1
	l_2	R_1
	l_3	R_2
g_2	l_4	R_3
	l_5	R_3

In addition to the payload which is encrypted with the group key, data packets during data transmission phase carry only the group address. Intermediate routers identify their next hops based on the group address and send only one copy of the data packet to each one.

4. *Join Reply (join_rep):* Once RP receives a join_req message with a valid token, it sends a join-rep message in the opposite direction (downstream) to the router from which it received this message. Each intermediate router in turn sends such a message to the downstream router from witch it received the relevant join_req for the particular 2^{nd}-leg-id. Failure to receive an appropriate join_rep after a timeout period causes a router to prune the corresponding 2^{nd}-leg-id from its MIB. A join_rep packet sent from R_i to $R_{(i-1)}$ to authorize the 2^{nd}-leg id *l* looks like the following:

$$\{\text{join\_rep}, E_{R(i-1)}(l)\}$$

5. *Leave Request (leave_req):* Similar to join_req in format sent by a subscriber to indicate its intention to leave a multicast group.
6. *Leave Reply (leave_rep):* Similar to join_rep in format sent by RP to authorize removing the corresponding subscriber from the MIBs of the intermediate routers.

III.9 SUMMARY

In this chapter V-routing protocol was proposed. This protocol provides communication anonymity and location privacy for both source and destination in a connection in ad hoc networks. In V-routing, the destination node is largely in control of what paths should be taken by packets that are destined for it. Packets are sent from the source to a transient destination named rendezvous point that has been selected by the destination. The rendezvous point forwards the packets to the destination over a hop-by-hop virtual circuit that is known only to the destination. Each node on this path only knows its next hop. We adapted V-routing to two classes of wireless infrastructure mesh ad hoc networks:

1) community networks where a client trusts its access router.

2) WISP network model where the client does not trust its access router but it trusts another currently remote router (e.g. its home agent in a MIP model) in terms of its identity but not in terms of its location.

Another flavor of V-routing was proposed for situations where the client does not trust any other node in the network. We applied our general concept of *destination-controlled routing* to design V-routing protocol for these network models.

We discussed the performance and the security of V-routing in presence of a global eavesdropper as well as proposed a public key certificate management system to

be used with V-routing. Also, we took advantage of V-routing for designing an anonymous multicast routing protocol.

The path establishment procedure of V-routing described in this chapter may not scale for very large networks. For example, every node or their access router must maintain a database of rendezvous points used by other nodes. A hierarchical architecture could provide better scalability. Extensions to this protocol may include topology disclosure protection (location privacy for routers) and QoS support. Specifically, QoS may depend on the selection of VHRs. An important part of any routing protocol is a maintenance protocol. In our protocol, one important case of maintenance is repairing the second leg, in a secure manner, when the access router of the destination changes.

One must make sure that this design for anonymous communication can safely integrate with other essential elements of a network such as accounting, billing, etc.

CHAPTER IV

DCARPS

In this chapter, we apply DCR in the context of wireless sensor networks. We will propose Destination Controlled Anonymous Routing Protocol for Sensornets (DCARPS). We will show that when the network load is sufficient, DCARPS provides location privacy and anonymity for the sink at all times. It also provides these properties for the sensors during data transmission. However, depending on the topology discovery protocol options, it may or may not support these features for sensors during network topology discovery phase.

Recent technological advances in the field of *Wireless Sensor Networks (WSN)* in terms of hardware and software hold the promise of continuous and accurate access to knowledge about our environment and events that happen in our surroundings as well as remote areas. However, wireless sensors are characterized by limitations on batteries, computational capabilities (CPU power and memory), as well as communication capabilities such as bandwidth, transmission power and receiver sensitivity. The nature of sensors and the way they are deployed make the re-supply of these devices with new energy resources highly impractical, and in many cases impossible. The lifetime of a sensor network depends on the amount of energy stored in the sensors and on the level of efficiency at which this energy is consumed. Sensors have to often be deployed in large numbers, cover vast areas and are usually unrecoverable. Thus, it is essential that their

cost of production be low. In addition, their size is expected to be small. These constraints place significant limitations on the computational and communicational capabilities of these devices.

In the early days of wireless sensor networks, the issue of network security was given second priority as the technology struggled to meet these strict and diverse constraints. It is only recently that a flurry of activities has been seen in some areas of wireless sensor networking including lightweight cryptography, secure routing and intrusion detection. However, *anonymous routing* that covers areas such as node anonymity, node location privacy, untraceability and unlinkability (defined in chapter I) is yet to be adequately explored. These issues are of utmost interest in military, homeland security, and law enforcement but are also becoming increasingly essential to many civilian applications.

Sensors are often deployed in an ad hoc manner, often in large numbers and over geographically wide areas. A typical deployment method for these networks, especially in the case of very tiny devices (e.g. dust sensors), is to spread them randomly out of an aircraft or a moving land vehicle. As a result of such a random and uncontrolled deployment process, wireless sensors normally form networks whose topology is unknown at the setup time. Moreover, due to its ad hoc nature, during its lifetime, a WSN usually possesses a dynamically changing topology that may be caused by battery-drained nodes, node/link failures, nodes joining/leaving and in many cases node mobility. Because of these issues, anonymity schemes such as Mist Routing [56] that assume fixed network infrastructure are not applicable to sensor networks. Also, most anonymous protocols designed for generic ad hoc networks, some of which were described in chapter

II (e.g. ANODR) and our own V-routing, are not suitable for wireless sensor networks due to their particularly limited resources.

In this chapter, we propose a new anonymous routing protocol for WSN based on our DCR approach. This protocol hides the identity and the location of the sink (the control center in a WSN) as well as supports source anonymity and location privacy. In this protocol, the sink as the sole destination node in a WSN determines the routes that packets from various source nodes must take in order to reach it. However, this is done in a way that the location and the identity of the sink are not revealed. In fact, by eliminating the common sink-originated beacons, we propose a new approach to network topology discovery that allows the sink to obtain a global view of the topology and thereafter establish data delivery paths based on label switching.

In the next section, we describe the usual method of routing packets in a WSN which is based on sink-originated beacons and we will explain a security vulnerability of this method. In section IV.2, we mention some special characteristics of WSN and how we can use them to our advantage. In section IV.3, we discuss the importance of location privacy and anonymity in WSN and introduce anonymity concepts in this context. In section IV.4, we categorize adversaries in terms of physical coverage. In section IV.5, we divide networks based on their traffic models and specify which traffic model can be supported for each adversary model. In section IV.6, we describe our assumed network model. In Section IV.7 we provide a detailed description of DCARPS. In this section, we also propose and evaluate a new network topology discovery protocol that can be used to support our new routing protocol. We also, propose a spanning tree construction algorithm for WSN that promotes fairness and load balancing. In section IV.8, we

evaluate the security offered by our proposed protocol using a stochastic approach as well as a simulation study. In section IV.9, we analyze the performance of our protocol in terms of overhead. And finally, we will conclude with a summary.

IV.1 ROUTING IN WIRELESS SENSOR NETWORKS

Wireless sensor networks usually consist of many small devices called *sensors* that are deployed in an area to collect different kinds of information such as environmental parameters, movements, habitual events and so on. Normally, the sensors periodically or by request send their readings to a special node in the network called the *sink* (a.k.a. base station, controller and processor). However, because of the usually small transmission range of sensors (due to their energy and size restrictions), data from distant nodes must be delivered to the sink using multi-hopping. Therefore, a multihop routing protocol is needed to find the most appropriate path from the source to the.

In the case of slow-changing topologies, typical of most sensor networks, a proactive routing approach seems efficient [119]. Although, as discussed in chapter II, the efficiency of proactive routing protocols in mobile networks has also improved significantly recently. As in all proactive routing protocols, a Network Topology Discovery Protocol (NTDP) is needed in order to acquire a global view of the entire network topology. To the best of our knowledge, most of the routing protocols designed specifically for this kind of network [119] base their NTDP on a special form of distance vector approach (a proactive approach) in which the sink usually broadcasts a periodic beacon in the network. This beacon carries a hop-count which is incremented every time it is relayed by a sensor. This way, each sensor can find out its distance from the sink through different directions and based on that it can select one of its neighbors as

its *parent* (upstream node that relays data for it towards the sink) that provides the shortest path. This way, a shortest-paths tree rooted at the sink is configured over which all packets would be transported to the sink.

It can be easily seen that, while resource-efficient, this approach reveals the location of the sink in the network to other nodes and potential eavesdroppers. This method is used even by those protocols that are concerned with anonymity and location privacy e.g. [79- 82, 119]. This design has been probably motivated by the desire to conserve the limited amount of energy and bandwidth in sensor networks, giving security a lower priority. However, there are many situations where security is at least as important as efficiency, for instance, in temporary sensor networks sometimes used in law-enforcement or military operations. In these situations, security is of utmost importance, while battery conservation may not be a big issue. As an example, consider the scenario depicted in Figure IV-1 in which the police and criminals are both looking for a valuable object that is acting as a source node emitting periodic beacons signaling its location. The police are using a multihop WSN to locate this object but the criminals can also use this setup to track down the source node as well as the police patrol in order to avoid them or even eliminate them. Periodic beacons received from the police reveal how far away they are while hop-by-hop packet tracing can lead the criminals to the source and/or the sink (the police). The importance of location privacy and anonymity in this example is obvious but the short-term objective of this network indicates that energy conservation is not the primary concern. There are also many applications of sensor networks where the nodes may not have serious energy constraints (e.g. video surveillance where the nodes may not be powered by batteries) while hiding the location

of the sink is very important as an intruder can render the entire system ineffective by removing this node.

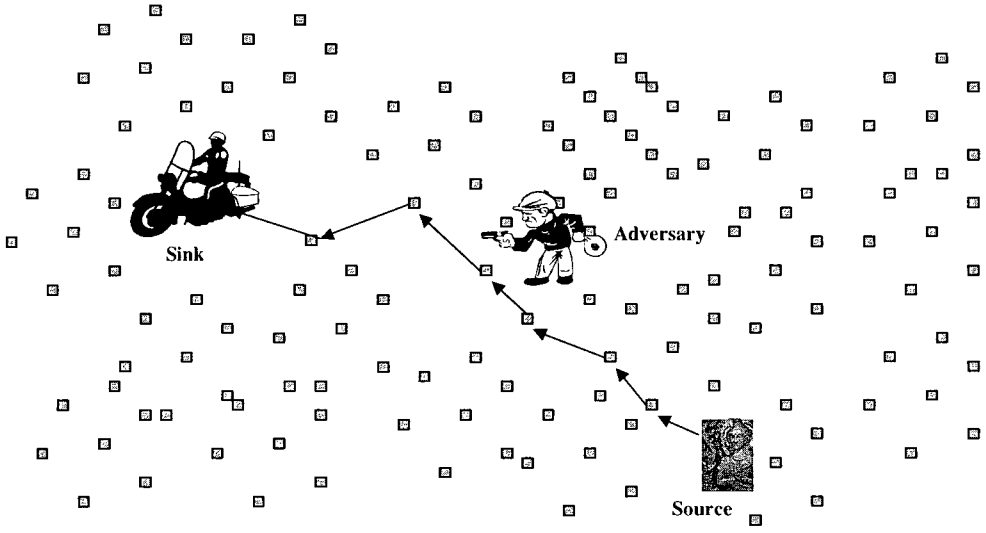


Figure IV-1 Importance of location privacy in a WSN

In section IV.8, we will introduce an anonymous routing protocol for WSN that takes advantage of a novel NTDP proposed by us, which preserves sink's location privacy during topology discovery. This protocol does not use sink-originated beacons.

IV.2 SPECIAL CHARACTERISTICS OF WSN

An important difference between sensor networks and other kinds of wireless ad hoc networks is that in sensor networks there is only one node (or just a few) as the destination node for all data flows. In other words, a sensor does not send its data to any arbitrary node in the network but it sends it only to a sink. In a small network of sensors,

there may be only one sink while in more complex networks multiple sinks may be present. This characteristic of sensor networks makes the routing for them simpler than general-purpose ad hoc networks because a sensor needs only to be able to send packets to one or a few special nodes as opposed to any node in the network.

Another special characteristic of sensor networks compared to generic ad hoc networks is that usually the sink is more powerful than other nodes in terms of energy and computations. Therefore, assuming that sensors can receive and interpret commands from the controllers, we can design a simple routing protocol in which the sink calculates the best routes for all the sensors and then informs them about these routes. This way, the computational burden of sensors, which are normally restricted in terms of energy and processing power, is reduced to a large extent because only the sink needs to have a global view of the network topology. In this chapter, we will propose such a routing protocol based on label switching that also supports location privacy and anonymity. This approach makes the key management system simpler as well since the sensors and controllers can be pre-programmed with shared secret material.

IV.3 MOTIVATION FOR ANONYMOUS ROUTING IN WSN

Source Location Privacy: This service is particularly needed for providing security to sensors in covert operations such as military, surveillance and law enforcement scenarios in which the sensors are vulnerable to capture and destruction. Another example is a stationary network of sensors that monitor movements of a valuable object, for instance in habitat monitoring applications. In this case, the object being tracked is the source and hiding its location from adversaries may be of interest.

Destination Location Privacy: Concealing the location of the sink(s) in a hostile environment is often the most important security issue in a sensor network. It is desirable to hide the destination from adversaries as well as internal nodes because they may be compromised. The sensitivity of the destination location is sometimes much more than sensors because the destination in sensor networks is the controlling entity and the central processing unit in the network. In sensor networks deployed in hostile environments, the location privacy for the destination is often more important than source location privacy, because an enemy can more easily destroy one sink node (collecting the information) than many sensors. Therefore, we are particularly interested in keeping the location of the sink a secret while retaining the capability to deliver packets to it.

Communication Anonymity: The importance of this property is also best explained with an example. Imagine that a large number of sensors of various kinds have been deployed in a large area (e.g. a city) collecting different kinds of data (e.g. temperature, chemicals, motion, traffic volume, ...). There are also many consumers (companies, organizations, ...) making use of the collected data. However, the fact that a certain consumer is collecting a certain kind of data may be sensitive information. Therefore, consumers may like to obtain sensors' data anonymously. An anonymous routing protocol must ensure that a packet cannot be traced back to both the consumer and the sensors used by it. In other words, at least the identity of one of the consumer and the sensor must be hidden.

Packet Tracing Attack

An anonymous routing protocol for WSN must prevent an adversary from finding the locations of the source and the sink. The adversary may exploit information in the packet headers or perform packet tracing. The former problem can be addressed using encrypted packet headers or identity-free routing, while the latter must be addressed by untraceable routing schemes. Encryption is undoubtedly one of the most effective means of providing security services, however it is not enough for ensuring anonymous communications. Adversaries can use *traffic analysis* techniques such as content analysis and timing analysis to mount packet tracing attacks in order to uncover the locations of the source and the destination of each packet as well as the identities of the nodes involved. Untraceability refers to the inability of an adversary in tracing individual data flows back to their origins or destinations. As was explained before in chapter I, packet tracing attacks can be done using physical layer techniques such as triangulation, tri-lateration, angle-of-arrival and signal strength detection in order to find the immediate sender of a packet at each hop. By following one or more packets belonging to the same data flow in this manner, an attacker can locate the origin and/or the destination of that data flow.

DCARPS is a novel routing scheme for sensor networks based on label switching and our own DCR approach that supports traffic untraceability in order to hide the locations of a source sensor and the sink. It is also an identity-free routing protocol and as such offers anonymity to the source and the sink. The identity-free routing and untraceability ensure unlinkability even when multiple sinks use the same sensor network.

IV.4 ADVERSARY MODELS

Location-limited Adversary: Such an adversary (also referred to as *local*) has a limited hearing range and can only monitor transmissions happening in its vicinity. It can track down the source or destination by gradually tracing a continuous stream of packets coming from a source. With the reception of every new packet it moves one step towards its target. It is usually assumed that the hearing range of this attacker is at least as big as sensors'.

Omni-present Adversary: Such an adversary (also referred to as global) can monitor all transmissions any where in the network and at all times. This adversary model can be implemented, for example, by a set of collaborating local adversaries scattered across the network. This is a much more powerful adversary that may even be able to identify the source or destination using only one packet transmission as opposed to a continuous stream of packets. In fact, the same packet can be observed over time by multiple adversaries who share and correlate their information. These collaborators may share their findings immediately, using a high speed communication channel.

IV.5 TRAFFIC MODELS

***Scarce Traffic*:** In this kind of network, only one or a few (non-overlapping or intersecting) data flows exist at the same time. In other words, only one or a few sensors are actively sending data at a time. Such a traffic model can resist a location-limited attacker at best because it does not have a sufficiently large anonymity set in order to confuse a global adversary. A global adversary can identify the source of a packet in such

a network as soon as it is generated because it does not mix in with any other flows. A local adversary can mount a packet tracing attack to uncover the source or the destination if a data flow continues as a steady stream of packets, what we refer to as *fixed-path* routing. Randomized routing techniques, such as our probabilistic DCARPS protocol, introduced in the next chapter can support location privacy and anonymity for this kind of traffic against local attackers.

***Abundant Traffic*:** In this kind of network, many sources would be sending data at the same time. Multiplicity of sources makes supporting this kind of network a simpler task. With a sufficient pool of active data flows (a large anonymity set) and with the appropriate techniques to make them indistinguishable, this traffic model can be protected even against a global adversary. In this situation, a packet being transmitted by a node might be an original packet or just a repeated one. Also, finding the right path to follow in a path tracing attack is more difficult for an attacker because at any relaying point s/he may have to choose one of many possible next hops. In section V.8, we will propose the basic version of DCARPS for this kind of traffic against a global adversary.

IV.6 NETWORK MODEL

We envision a network of many (hundreds or thousands) small wireless sensors that are deployed in a large geographic area. We also assume that a single, considerably more powerful node, called *sink,* exists in the network that controls the sensors and collects

their readings[7]. Because of their limited transmission range, sensors send their data to the sink using multihop communication. Sensors generate data independently and relay packets received from their neighbors without deterministic knowledge of their arrival times. In other words, we are not assuming node scheduling. Packet generation according to a node scheduling mechanism is more vulnerable to timing analysis attacks, because of regular timing relationship between the nodes' transmissions. In our network model, a node is always ready to receive a packet from its neighbors and retransmit them according to a certain policy. Some energy-efficient MAC protocols (e.g. SMAC [120]) allow nodes to detect packets while in an idle mode. Data packets generated by sensors always travel upstream (uplink) towards the sink and are never addressed to any other node. Control packets such as routing updates from sensors also travel upstream, destined for the sink, while control packets from the sink such as routing instructions, cryptographic assignments, acknowledgments and so on, travel downstream (downlink). We assume bi-directional links only; meaning two nodes are considered neighbors if and only if they can hear each other.

In accordance with the DCR paradigm, our approach to ensure anonymity and location privacy for the sink is to avoid disclosing information about its identity and its location as much as possible. To this end, we have made the sink entirely in charge of how packet routing should be done. Through a topology discovery process, described later, the sink obtains a global view of the network topology, but no other entity knows which node is the sink. We will also explain how it can securely and anonymously send

[7] The sink may also be a sensor but we use the term sensor to refer to all the member nodes other than the sink.

instructions to each node in a manner that ultimately all packets can find their way to the sink, without the privy of any internal or external entity.

The sink is assumed to have sufficient energy and processing capability to calculate all the necessary routes (a reasonable assumption, considering that the sink could be an access point linked to a laptop or a PC, or connected to the Internet). Obviously, this centralized approach raises several concerns such as single point of failure and scalability. The former problem can be dealt with using redundant sinks, while the latter can be resolved using a hierarchical architecture. Besides, routing is not the only function of the sink and the whole network operation depends on the survivability of this important node. Sensors have modest computational (for symmetric cryptographic operations) and storage capabilities. For example, a node of type US Berkley Mica mote [121, 122] is capable of performing RC5 encryption/decryption in reasonable times [880] and has 128 KBytes of flash memory on its Atmel Atmega-128 microprocessor.

IV.7 PROTOCOL DESCRIPTION

In this section, we describe the basic version of DCARPS (Destination Controlled Anonymous Routing Protocol for Sensornets) for an abundant-traffic situation. This protocol consists of 6 parts; *Initialization, Topology Discovery, Route Calculations, Paths Establishment, Uplink Data Transmission and Downlink Data Transmission.* In the next chapter, we will extend DCARPS to support scarce-traffic wireless sensor networks.

IV.7.1 Threat Model and Privacy Objectives

We intend to resist an omni-present adversary. This kind of adversary can monitor all transmissions happening anywhere in the network. It can theoretically follow one single packet to its destination. It can also detect the source of a packet as soon as it is generated. Therefore, in order to have an acceptable degree of protection against this kind of adversary, we need two things: 1) make packets indistinguishable 2) have a sufficient number of active sources in the network. This way, we can make the process of distinguishing a specific flow among many flows difficult. The greater the number of data flows (anonymity set) the higher the degree of anonymity will be. While this condition is intrinsically satisfied in many types of sensor networks such as inventory tracking, environment monitoring and some habitat monitoring applications, in some situations it may have to be artificially ensured via fake traffic. Also, given a specific packet and the entire transmission history of the network, we would like to make the task of identifying the sender of the packet as difficult as possible for an eavesdropper. Finally, and most importantly, we wish to hide the location and the identity of the sink.

Later, we will explain how packets belonging to different flows can become indistinguishable. Thus, we assume that as far as the eavesdropper is concerned, a packet transmission by a node may be for an original packet or just a relayed packet. Sensor nodes are usually vulnerable to capture and intrusion. An adversary is assumed to gain complete control of a compromised node including its routing information, collected data and all its cryptographic material. Therefore, it is important that no node is informed of the locations and identities of the sink or other data sources. As in V-routing, we are only concerned with eavesdropping actions (passive attacks) of an adversary who is trying to

locate a target node. Such an adversary usually prefers to be hidden, thus it refrains from executing active attacks e.g. denial of service, impersonation, jamming and so on, which may be discovered by the network operator using intrusion detection techniques.

IV.7.2 Various Stages of the Protocol

Part 1: Initialization

Before being deployed, a node *i* (including the sink) is assigned a unique network identifier S_i. The sink and a sensor S_i are pre-programmed with a unique shared secret key K_i. During the lifetime of a sensor, its key may be updated by the sink. Due to computational and energy limitations of sensors, we only use symmetric cryptography[8]. Therefore, we do not require a Certificate Authority (CA) nor do we assume any key management system such as KDC. Prior to deployment, the sink also shares a value with each sensor, denoted by DI_i (for Downstream Incoming), whose use in downlink communications will be explained later.

Part 2: Topology Discovery

DCARPS requires the sink to have a global view of the network topology. Thus, it needs a network topology discovery protocol (NTDP) as discussed before. Although we do not dictate what NTDP must be used, we require this protocol to preserve the anonymity and the location privacy of the sink. Therefore, it cannot use sink-originated beacons. In this section, we propose one such NTDP that is based on a completely different and

[8] Using light-weight cryptography, asymmetric cryptography on sensors is now possible as well.

distributed approach. Instead of emitting beacons from the sink, some or all of the nodes (at the worst) broadcast their identity and let the sink, passively and anonymously, obtain a global view of the network topology. In fact, cumulative *Route DIScovery (RDIS)* messages originated at sensors are flooded throughout the network, gather the identities of the sensors that they meet on their way and deliver them to the sink. This method is similar to the first pass of the route discovery process in DSR (RREQ), but it only allows the destination (sink) to learn the path to a source node. This design is contrary to the common method of topology discovery in sensor networks based on broadcasting beacons from the sink. As explained before, that method violates location privacy of the sink. By making all the nodes, including the sink, behave in the same manner during topology discovery, we have effectively hidden the sink. We refer to this protocol as the sink-anonymous NTDP. Later on, we will extend this protocol to also preserve the secrecy of the network topology during the topology discovery phase with regard to every entity besides the sink. We refer to this extended version as the all-anonymous NTDP because it hides the location of all the nodes.

Sink-Anonymous NTDP

The goal of this protocol is to let the sink learn the relative positions (not necessarily the geographical coordinates) of all the sensors in the network. This process is repeated periodically within *topology discovery rounds*. Furthermore, we divide a topology discovery round into multiple *check-in* periods. Please see Figure IV-2. At the beginning of a check-in period, a node that has not yet originated or repeated an RDIS packet within the current round of topology discovery, wakes up and decides whether it should generate

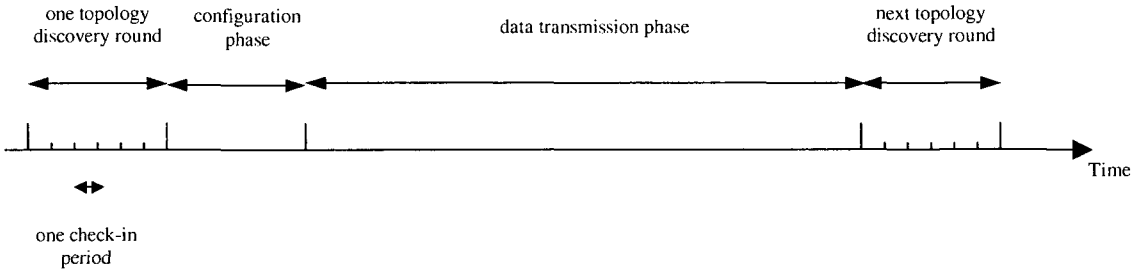


Figure IV-2 Different operational phases of the WSN

a RDIS packet. It makes this decision according to a Bernoulli random variable, which describes an experiment with a simple "yes or no" outcome. In other words, when such a node wakes up, it generates an RDIS packet with probability p and it does not with probability $(1-p)$. RDIS packets are distributed within the network using controlled flooding which means an intermediate node forwards only one copy of such a message and discards the other copies. In order to prevent adversaries from identifying the sink based on a special behavior, the sink follows the same rules as the other nodes, as far as the generation and the forwarding of RDIS packets is concerned.

An RDIS packet consists of a globally unique sequence number and a variable-length field called the *route* field. A globally unique sequence number can be generated using hash functions.[9] The originating node of a RDIS packet, and afterwards its forwarding nodes, insert their IDs in the route field of the packet. The maximum length of the MAC frame dictates the maximum length of the route field and hence the diameter of the network. For very large networks, a hierarchical architecture may be employed. As an example, Figure IV-3 depicts part of the process of route discovery initiated by one

[9] As mentioned before , IETF RFC 4122 defines a namespace for globally unique identifiers.

of the nodes, S_1. In this figure, and in the rest of this chapter, we denote the unique network identity (ID) of a node i with S_i. The format of an RDIS packet is also clear in this figure.

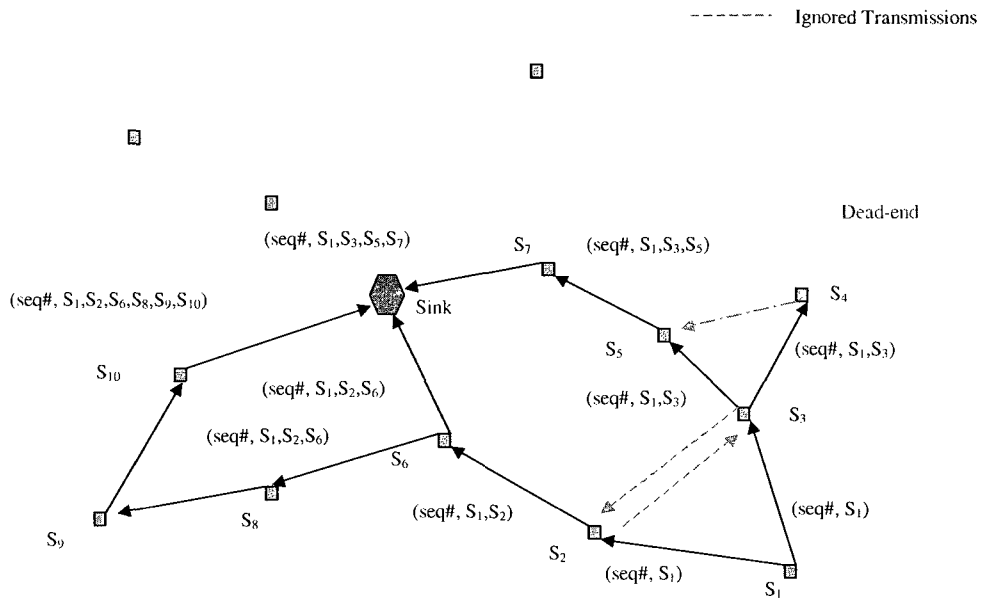


Figure IV-3 Topology Discovery; S_1 issues an RDIS

In order to limit the amount of control traffic, a node refrains from generating an RDIS packet if it has already relayed such a message from other nodes. The sink can identify and locate a sensor as long as it receives at least one RDIS packet containing the ID of that sensor. We further limit the control overhead of the protocol by using limited-range flooding, which means the maximum distance that a flooded message is allowed to travel is restricted. Therefore, an RDIS packet contains also a Time-To-Live (TTL) field witch is initialized to some value and is then decremented every time the packet is repeated. An RDIS packet is discarded when its TTL value becomes zero. A reasonable value for TTL may be the diameter of the network to make sure that all of the sensors can be discovered

by the sink, if the sink happens to be located at the edge of the network. In the following section, we provide one forwarding scheme for RDIS packets that ensures the controlled flooding of these packets.

A forwarding node repeats an RDIS packet with a certain global sequence number only once. Future copies of the same message are discarded by the sensor. An RDIS packet may fail to discover all sensors in its coverage area due to different reasons including network partitioning, TTL deadends and the fact that nodes only repeat the first copy of the packet. Therefore, more than one such packet may have to be generated in one area. The result of controlled flooding is that from the perspective of the sink, different branches of a route discovery tree rooted at a sensor can only merge at the sink.

In order to guarantee that all of the sensors are eventually discovered, and to speed up this process, we employ three techniques:

1. We ensure that, at the beginning of a topology discovery round, some of the nodes issue RDIS packets with probability 1 (*starter* nodes). Obviously, it is desirable for these nodes to be positioned at a suitable distance from each other so that their coverage areas have minimal overlaps. During the lifetime of the network, the sink knows the topology of the network and can make intelligent decisions about the next round of topology discovery in terms of selecting the nodes that are to initiate RDIS packets.[10] The outcome of this selection process can be conveyed from the sink to the sensors within the body of downstream control packets. The implementation of these control messages is addressed later in this section but we just point out here that

[10] While not essential for the successful operation of our protocol, the optimization of this selection process can enhance its performance.

these messages are indistinguishable from data packets generated by sensors and do not reveal the identity and the location of the sink.

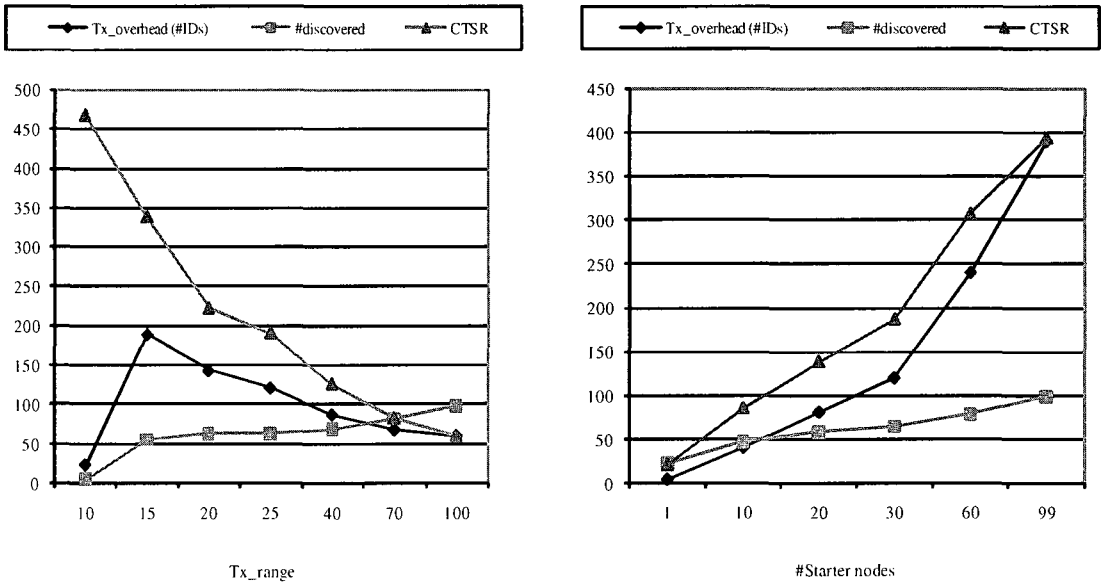
2. At the deployment time, there is little control over the placement of each sensor and thus it is very difficult to achieve an optimal distribution for the RDIS issuing sensors. However, we still envision that sensors are deployed in batches and that each batch covers a limited geographical area. Then, we can assume that at least one sensor per batch is pre-programmed to initiate a route discovery message some time after activation.
3. A time threshold towards the end of the topology discovery round may be considered, which if reached before a sensor generates or repeats an RDIS packet, causes that node to immediately originate a route discovery message.

Even if a node generates an RDIS packet or repeats RDIS packets from other nodes, there is a chance that it may not be discovered because those messages may reach one of several kinds of deadends mentioned before. In that case, this node will be missing from the downstream path setup packets because the sink is not aware of it. In case of that happening, this node will generate an RDIS packet at the beginning of the next topology discovery round with probability 1.

Performance Analysis of Sink-anonymous NTDP

In order to evaluate the performance of our NTDP, we implemented it using the OPNET simulation software in a 100x100 area using 100 stationary nodes. We experimented with different values for the transmission range of a node (Tx_range), number of starter

nodes (#starters) and the probability of RDIS generation (p). In each case, we measured the mean transmission overhead per sensor in terms of the number of IDs that are transmitted as well as the sequence numbers (assumed the same length as an ID). We also observed what percentage of nodes were discovered in each case. Then, we defined a metric called *cost to success ratio* (CTSR) as the ratio of the total transmission overhead to the number of discovered nodes. Obviously, we would like the CTSR to be as small as possible. In all of our scenarios, the sink and all of the sensors including the starters are randomly positioned in the network. We considered TTL=25, which was never reached.



(a) (b)

Figure IV-4 Performance of sink-anonymous NTDP

Figure IV-4a shows the transmission overhead per sensor, the number of discovered nodes and the CTSR for different values of transmission range when there are 30 starter nodes and p=0.3. From this figure, we can see that when transmission range is smaller than 15, many nodes remain undiscovered. However, this is not a limitation of our protocol and is the result of poor network connectivity. In fact, under these circumstances, the network is a collection of isolated islands of nodes and the sink is only capable of discovering the nodes within its own island. For values of Tx_range greater than 20, the network seems to be well connected but still many nodes are not discovered due to the effect of controlled flooding because all of the RDIS messages that they relay are dropped on the way to the sink. This happens usually when a group of nodes is connected to the rest of the network with fewer links than the number of nodes. For Tx_range greater than 70, most of the nodes are quickly and with little overhead discovered because they are direct neighbors of the sink. However, this range should not be our focus because such a network does not constitute a real multihop network and does not need topology discovery or routing protocols since a sensor can send its data directly to the sink. A reasonable sensor network would probably consist of nodes with transmission ranges in the 20-30 range. Commercial sensor devices have a transmission range in the order of a few meters or tens of meters. For a sensor with an actual transmission range of 30m, Tx_range=20 means that our network spans a 150m x 150m area.

Figure IV-4b shows the same graphs for different numbers of starter nodes when Tx_range=25 and p=0.3. As can be seen, all of the nodes are guaranteed to be discovered if each one issues an RDIS packet. Unfortunately, the amount of communication

overhead increases drastically when the number of starter nodes is greater than 10. However, with 10 starters, only half of the nodes are discovered by the sink. This observation suggests that instead of increasing the number of starter nodes, we may be able to discover all or most of the nodes if we invoke the topology discovery process two or three times, every time using a different set of 10 starter nodes. Modifying the controlled flooding behavior of the protocol, such as allowing each node to repeat the same sequence number twice, may also help improve the performance of the protocol. However, partial discovery of nodes in one round of topology discovery may not necessarily be a disadvantage. Due to random distribution of sensors, quite often, nodes in a WSN are too densely deployed in some areas. In that case, a group of sensors located too close to each other report similar and redundant data. This phenomenon results in the waste of energy and bandwidth. When only a subset of nodes issue RDIS packets, our NTDP protocol allows the sink to discover only some of the nodes in such a group. The rest of the group will remain inactive during the following data transmission phase, enhancing the longevity of the network and reducing the control traffic overhead. Ultimately, the number of starter nodes can be used as a protocol parameter in order to balance the overhead and the population density of the active nodes.

In our experiments, while the network is reasonably connected, changing p does not seem to affect the performance, since most of the nodes either issue or repeat an RDIS packet within the first or the first few check-in periods.

While the transmission overhead of our protocol may seem too high compared to the protocols based on sink-originated beacons, we believe that it is still well within the capabilities of today's sensors and is also justified considering its security benefits. In

order to put things in perspective, we use some data reported in [113] regarding the commercially available Berkley MICA motes to estimate what is possible on a common sensor device. According to this paper, 2 AA batteries provide almost 2200mAh to these sensors and they consume almost 20nAh for a packet transmission, 8nAh for a packet reception and 1.25nAh/ms for radio listening. In our experiments, we found that with 100 nodes, when Tx_range=25, each node has 15 neighbors on average. So, we can assume that one packet spends almost 150nAh to be received by all of the neighbors of a sender. Therefore, it is safe to say that the transmission of an RDIS packet and its reception by neighbors consumes almost 170nAh (by 16 nodes) or 10.6nAh per node. According to the afore-mentioned paper, almost 25% of the available energy (i.e. 550mAh) is at our disposal for the purpose of communications. Thus, during the lifetime of the two batteries, a sensor can handle 51,886,792 packets or 1,556,603,760 bytes, considering that the typical packet length for the TinyOS, the operating system of MICA motes, is 30 bytes, although it can support packet lengths of up to 128 bytes. We would like to note that in our experiments, the maximum length of an RDIS packet was always below 30 bytes but TTL can be adjusted to account for this limitation, if needed. If we assume that each ID is represented by 2 bytes, in the worst-case scenario (when every single node issues an RDIS message), our NTDP requires each sensor to handle less than 800 bytes during a round of topology discovery. With 10 starter nodes, this value is reduced to less than 80 bytes per node.

The viability of our protocol depends largely on the nature of the WSN and its application. Indeed, it directly depends on the frequency of the topology discovery rounds and the amount of user data that the network carries during the data transmission phase.

When the topology of the network is static or changes slowly, topology updates need to be done less frequently. Also, in many scenarios, such as real-time applications (e.g. video surveillance), sensor networks could carry substantial amount of data during their operational phase compared to the control traffic such as our topology discovery overhead. This means that the overhead of our topology discovery is justifiable by the amount of useful data that can be transmitted in the phase following it. Moreover, it is worth mentioning that in many applications of sensor networks, each and every data packet is flooded throughout the network while we use this technique only during a short period of time.

We would also like to point out that there are other protocols for multihop sensor networks that also need a global view of the network topology e.g. topology control and network management mechanisms [124-127]. These protocols allow the sink to monitor, control and co-ordinate node activities in the network in a manner that maximizes the overall network lifetime. Therefore, often, our NTDP is in fact using the information already collected by these protocols and does not introduce additional overhead. However, in our protocol, the sink needs to also broadcast path setup messages, whose overhead must be taken into account. The difference between our NTDP and others such as those used in [124-126] is that it can also be extended to support all-anonymous topology discovery because it does not require a node to identify itself to its neighbors.

All-Anonymous NTDP

Although our sink-anonymous NTDP hides the location and the identity of the sink during topology discovery, it does not preserve the secrecy of the network topology. In other words, it discloses the identities and the locations of all the nodes including the sink albeit without specifying its special role. This is adequate in situations where the individual identity of a sensor is not important, for example in monitoring and target tracking applications where a stationary network of sensors is used to detect the presence of other objects. However, there are many applications of sensor networks that require support against topology disclosure. For example, imagine a retirement residence, a hospital or a ski resort where all guests and the staff wear sensors and are actually part of the network. In such scenarios, the capability of a potential intruder to locate and identify a person at all times may be a real concern. In general, in those applications of sensor networks, where a sensor is associated with a real personality, topology disclosure may introduce a security risk. In this section, we extend our proposed NTDP to support fully anonymous topology discovery.

We propose two methods for maintaining anonymity and location privacy of sensors during topology discovery. These two methods can be, either individually or combined, applied to our sink-anonymous NTDP.

Method 1: Frequently Changing Pseudonyms

This technique is commonly used in anonymity systems to create temporal confusion for the adversary with regards to the identity of nodes. It simply changes the network names of nodes every so often. It can be more effective when combined with spatial confusion

of a dynamically changing topology, e.g. in mobile networks. The identifier of each node can change every topology discovery round or less often, depending on our security requirements. In our system, the strength of this technique in confusing an adversary is boosted by the fact that the set of RDIS issuing nodes is also changing every time. Therefore, correlating RDIS packets across different rounds of topology discovery is a more difficult task.

In our protocols, the sink and a sensor can independently calculate the next ID to be used by the sensor, using their shared key and a well-known one-way hash function. Privacy through using frequently changing pseudonyms has been analyzed to a great extent in the seminal work of Beresford on Mix Zones [127].

Method 2: Onion Cryptography

In this method, every node uses its end-to-end key to encrypt a route discovery message. First, the RDIS-issuing node includes its ID in this message, which is encrypted for the sink. Forwarding sensors use their end-to-end keys to add a layer of encryption to this cryptographic onion. For example, the copy of the route discovery message issued by S_1 in Figure IV-3, traversing the path $S_1 \rightarrow S_2 \rightarrow S_6$, will look like (seq#, K_6 (S_6, K_2 (S_2, K_1 (S_1)))) when it reaches the sink. K_i (x) denotes the encryption of x using K_i.

These two methods can be combined to achieve better security, as we will explain below. By using frequently changing node identifiers, we provide an extra degree of protection in networks with static topologies. In a static topology, if identities also remain unchanged, all route discovery messages will be repeated over consecutive topology discovery rounds. In other words, across an arbitrary link in the network, the route

discovery messages generated by a particular sensor will always be the same. This allows an adversary to acquire a vision of the network topology even without knowing the actual identifiers in each message as in the second method. As mentioned before, the rotation of the set of RDIS-originating nodes plays a complimentary role to this technique.

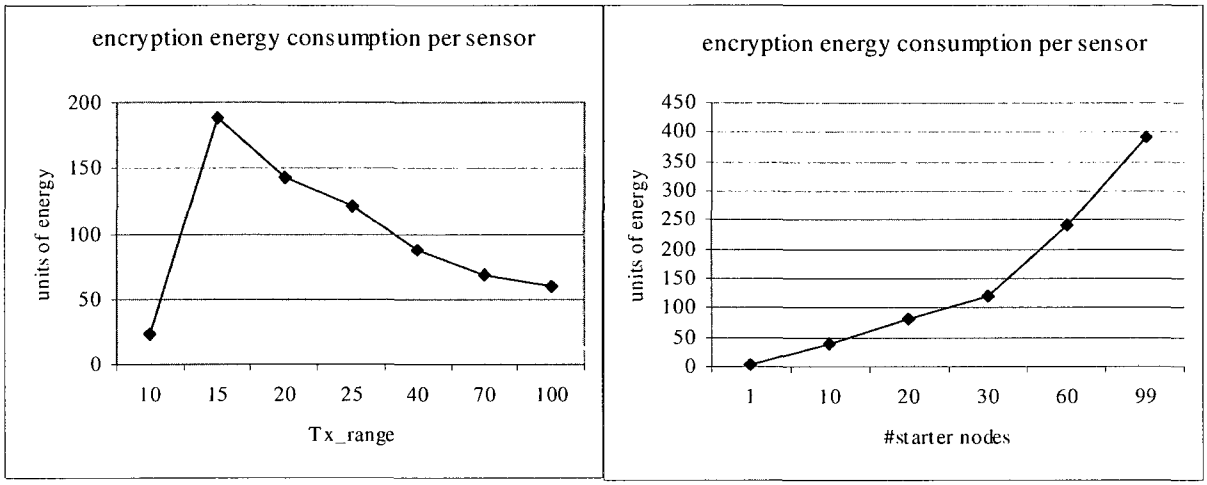
Changing the node identity from one topology discovery round to the next has the effect of a nonce in resisting against replay attacks for the second method. Otherwise, a replay attack against our protocol can be launched if an adversary retransmits an old RDIS packet after a while, with a new sequence number, in an attempt to produce stale routing information. If each node identifier is changed every round, the sink can keep track of route discovery messages and ignore the old ones.

Performance Analysis of all-anonymous NTDP

Our all-anonymous NTDP does not add any communicational overhead to our sink-anonymous NTDP. However, it does add some computational overhead as a result of its cryptographic operations. For the frequently changing pseudonyms method, the cryptographic overhead is small and depends on the employed hash function. For the onion cryptography method, if we ignore the decryptions performed by the sink, then the cryptographic overhead of our protocol is the result of layered encryptions done by sensors and is directly related to the transmission overhead of the protocol and the cryptographic overhead committed by a sensor.

The most important aspect of cryptographic overhead for a sensor is the energy consumption associated with it. Let's consider the energy needed to encrypt one ID as our energy consumption unit. In that case, figures IV-5a and IV-5b show the energy

consumption of our all-anonymous NTDP per node, during one topology discovery round.



(a) (b)

Figure IV-5 Cryptographic overhead of all-anonymous NTDP

Part 3: Route Calculations

Once the sink has acquired the network topology, it calculates routes for all the sensors. In the simplest case, it calculates only one route per sensor according to a pre-specified policy, such as "the shortest path". All the possible end-to-end routes for each sensor resemble multiple streams originating from the same point i.e. that sensor. In addition, even though each stream may branch out somewhere on the way but different branches of the eventual tree for each sensor can only merge at the final destination (the sink).

The final set of shortest paths is a spanning tree structure rooted at the sink. Each sensor may have multiple downstream links but only one upstream link towards the sink. In other words, a *main branch* (a link connecting the sink to one of its neighbors)

emanating from the sink may split several times into smaller branches, finally ending at individual leaf nodes. A *leaf* node is a sensor that does not have a downstream link on the shortest-paths tree. The shortest path for a leaf sensor contains the shortest paths for all its upstream sensors. A possible shortest-paths tree for the network of Figure IV-3 is shown in Figure IV-6. Please note that most wireless sensor networks use a similar routing tree.

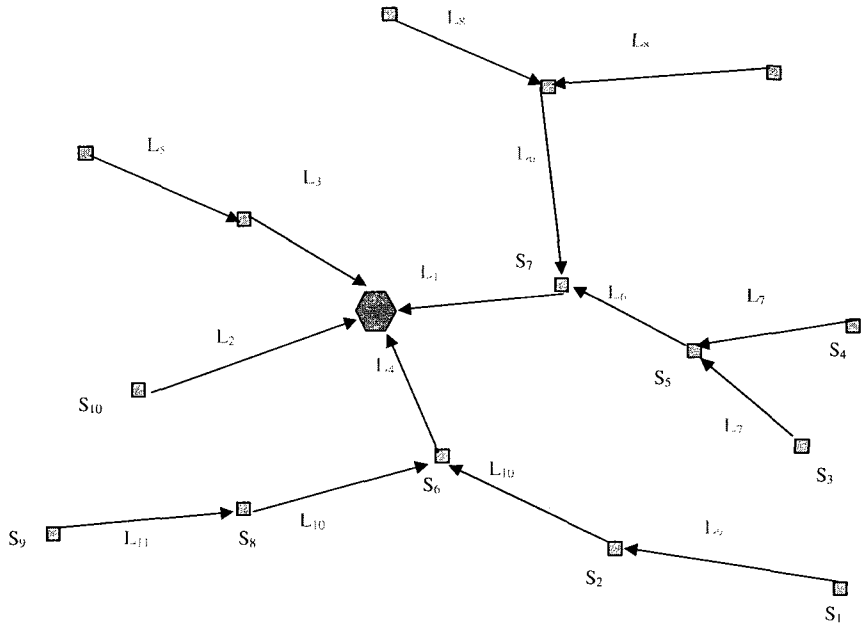


Figure IV-6 Label assignments in a shortest-paths tree

A Load-Balancing Technique

Having a global view of the network topology, the sink is able to calculate the shortest path between each sensor and itself. It can use any shortest-path algorithm such as Dijkstra's [128]. However, these algorithms have been designed to find the shortest paths

for a node to reach any destination in the network without considering the number of links that ultimately cross each node. In other words, they do not strive to distribute the links evenly. However, in the case of DCARPS, due to our unique situation in which the sink has all the data and control for routing decisions, it can also execute a fairness algorithm. In fact, it can try to equalize the number of children of each sensor. This feature is extremely desirable in wireless sensor networks because it enhances the overall lifetime of the network. A sensor with more children must repeat more packets for other sensors, which causes its energy to deplete faster. The lifetime of a sensor network is usually considered the duration of time that it takes for the first sensor to run out of battery.

In our fair shortest-paths-tree construction algorithm, the sink executes the following logic locally over its current network topology information.

Algorithm IV-1 Shortest-paths tree construction

Procedure *build_SPTree;*
(Input: N, $\{N^1(x) \mid \forall x \in N\}$, $SPTree = \Phi$)
(Output: *SPTree*)
Begin
For all x **in** N
    N'(x) = N^1(x);
    C(x) = Φ;
    x_assigned = False;
    x_processed = False;
SPTree = SPTree U{sink}; //Process the sink first.

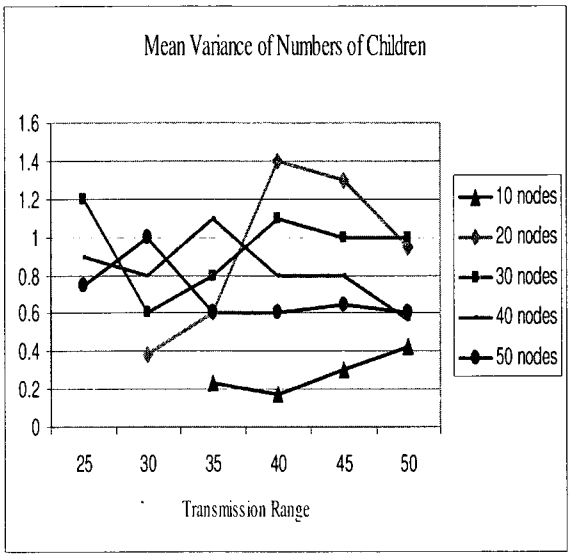
```
|SPTree| = |SPTree| +1;
Tier^1 = N^1(sink);
C(sink) = N^1(sink);
SPTree = SPTree U Tier^1;
|SPTree| = |SPTree| + |Tier^1|;
For all x in N^1(sink)
        x_assigned = True;
        x_parent = sink;
        N'(x) = N'(x) – {sink};
t = 0;
While SPTree ≠ N
        t = t + 1;
        Tier^(t+1) = Φ;
        no_processed = 0;
        While no_processd < |Tier^t|
                For all x in Tier^t
                        If (x_processed = False) Then
                                If (N'(x) ≠ Φ) Then
                                        C(x) = C(x) U {y};     // y ∈ N'(x)
                                        N'(x) = N'(x) – {y};
                                        y_assigned = True;
                                        y_parent = x;
                                        Tier^(t+1) = Tier^(t+1) U {y};
                                        SPTree = SPTree U {y};
                                        |SPTree| = |SPTree| + 1;
                                Else
                                        x_processed = True;
                                        no_processed = no_processed + 1;
End;
```

The notations of Table IV-1 have been used in this algorithm.

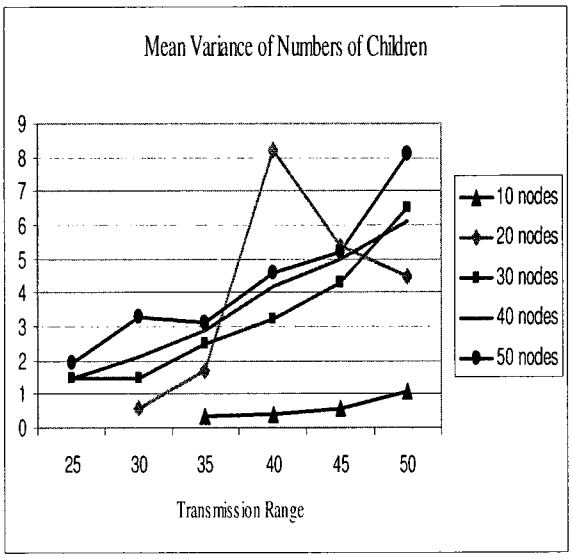
This algorithm runs in multiple rounds over consecutive *tiers* starting from the closest nodes to the sink. A tier is the set of all nodes that have the same distance from the sink. In each tier, the algorithm is executed in multiple iterations. In each iteration, each node in the tier is assigned at most one child from amongst its immediate downstream neighbors who do not have a parent yet. This way, if x and y are both upstream neighbors of two other nodes a and b, x will become the parent of one and y becomes the parent of the other one. The sink is automatically assigned as the parent of all of its neighbors. A tier is completely processed once all the nodes in the next tier have been assigned a parent from this tier. At that point, the algorithm moves on to the next tier. The algorithm terminates when all tiers have been processed and all nodes have been added to the shortest-paths tree.

Table IV-1 Notations

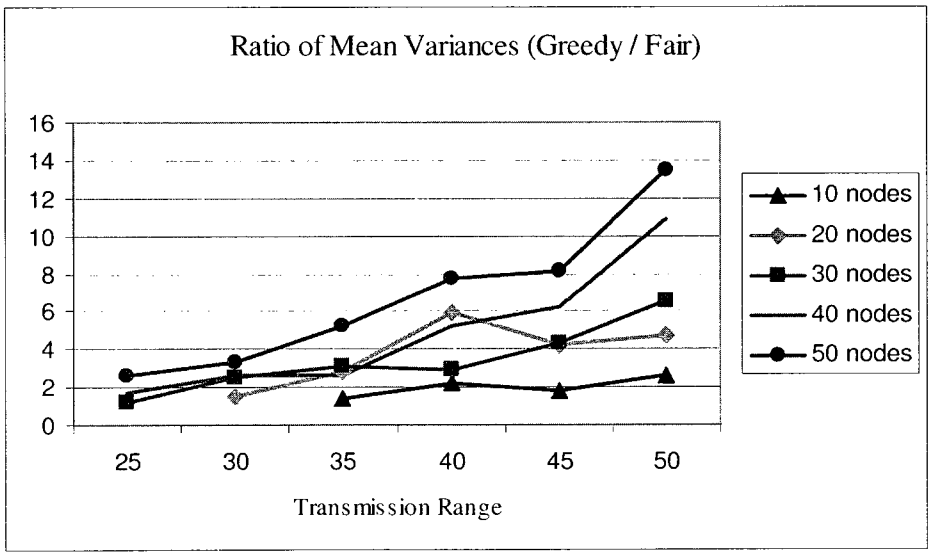
N = The set of all nodes including the sink	
x_assigned: True if x has a parent	//a Boolean value
x_processed: True if x has no neighbors without a parent	//a Boolean value
x_parent: x's parent node	
|A|: Cardinality (size) of the set A	
$\mathbf{h_x}$: x's distance from the sink in terms of hop counts	
$\mathbf{Tier^t}$: $\{x \mid h_x = t\}$	
$\mathbf{N^1(x)} = \{y \mid y$ is a neighbor of x.$\}$	
$\mathbf{N'(x)} = \{y \mid y \in N^1(x)$, y_assigned = False$\}$	
SPTree: $\{x \mid x \in N$, x's parent has been determined.$\}$	
$\mathbf{C(x)} = \{y \mid x$ is y's parent.$\}$	



(a) Fair



(b) Greedy



(c) Fair vs. greedy

Figure IV-7 Fair shortest-paths tree algorithm vs. a greedy algorithm

We evaluated the performance of our algorithm using the OPNET simulation tool. We used the variance of numbers of children of all the nodes as a metric in order to compare our algorithm against a greedy shortest-paths tree construction algorithm that assigns all unassigned downstream neighbors of a sensor as its children. Obviously, the smaller this value, the more the fairness of the algorithm. Figures IV-7.a and IV-7.b show this quantity for our fair algorithm and the greedy algorithm, respectively. We experimented with almost 150 different scenarios by varying the number of nodes, the transmission range (in a 100x100 area) and the network topology (random node placements). In every single case, our algorithm out performed the greedy algorithm as can be seen in Figure IV-7.c. This figure shows the relative performance of these two algorithms expressed as the ratio of the variance of number of children of the greedy algorithm and the corresponding value for our algorithm. As can be seen, this value is always greater than 1 which means our algorithm performs better.

Part4: Uplink Path Establishment

In DCARPS, the sink is responsible for establishing upstream next hops (parents) for all sensors. In this phase, the sink assigns labels for uplink communications to each node, including itself. A *label* is part of a packet that dictates how it should be forwarded. The label in an incoming packet at a node is called an *incoming* label, while the label in an outgoing packet is called an *outgoing* label. A node is the recipient of a packet if it has been assigned an incoming label that matches the label of the incoming packet. To forward a packet, a node swaps the label in the packet with its own outgoing label. Later, we will explain how the sink communicates these label assignments to sensors.

Due to the spanning structure of the routing tree, each sensor will have only one incoming label (even though it may have more than one incoming link) and one outgoing label in the upstream direction, because all sensors send their data to the same destination, namely the sink, and each one of them has only one upstream link. The outgoing label of a node is always the same as the incoming label of its upstream node. The sink may have no outgoing label, while the leaf sensors have no incoming labels. The sink can assign the same outgoing label to all of its neighbors. However, this may result in a large concentration of packets with the same label in its vicinity, which may be an indication to adversaries that the sink resides in that area. While some techniques such as transmission rate control and data aggregation may be used to equalize the packet density on all the links, we recommend a simpler solution for this problem which is assigning different outgoing labels to the neighbors of the sink. Using this method does not violate sink anonymity because it is still indistinguishable from most parent nodes in the network. In fact, every parent node in the network may have non-child neighbors that obviously use a different outgoing label than its children. This property is ensured by our load-balancing shortest-paths tree construction algorithm explained before. A label assignment example for a simple shortest-paths tree was shown in Figure IV-6.

Following a round of topology discovery, if the sink detects a change in the network topology (e.g. due to link failure, node outage or mobility) it initiates a new round of path establishment. With DCARPS, the sink uses special *path_setup* messages to inform each sensor of its assigned uplink incoming label and outgoing label. These messages travel downwards along the branches from the sink to each sensor, according to

the tree structure calculated previously by the sink. The format of these messages and how we maintain their secrecy is explained below.

Path Setup Message

In order to reduce the control traffic overhead, we distribute labels to all the sensors connected to one main branch by using just one message. The sink assembles a cryptographic onion per each main branch and broadcasts it locally. Each layer of an onion normally contains the incoming label and the outgoing label for the sensor that can successfully decrypt that layer. Each layer is encrypted by the sink using the secret key that it shares with the intended recipient. The layers are in such an order that one of the immediate neighbors of the sink can peel off the outermost layer of an onion. Each sensor broadcasts the rest of the onion after peeling off the outer layer. At each step, only one sensor can decrypt the outer layer. However, where a branch splits up, the corresponding sensor must broadcast as many sub-onions as the number of its downstream sensors. These sub-onions are prepared by the sink in a way that the outer layer of each can only be decrypted by one of the sensors downstream from the branching sensor.

In order to prevent unnecessary decryption attempts by those neighbors of a sender that are not the intended recipient as well as make the path_setup messages look exactly like data packets (to be indistinguishable for the sake of sink anonymity), we add a label at the beginning of each layer. This label is unencrypted and called a Downstream Incoming (DI) label. As we said in the initialization phase, each node is initialized with one such label before deployment. Only the node and the sink are aware of the DI label

for a particular sensor. As an example, the path_setup message broadcasted on the main branch labeled with L_4 in Figure IV-6 is:

$$[DI_6, K_6(L_{10}, L_4, [DI_2, K_2(L_9, L_{10}, [DI_1, K_1(-, L_9)])], [DI_8, K_8(L_{11}, L_{10}, [DI_9, K_9(-, L_{11})])])]$$

$K_i(x)$ denotes encryption of x using the shared key K_i. In the encrypted part of each layer, starting from the left, the first item is the incoming label and the next item is the outgoing label. Figure IV-8 illustrates this example.

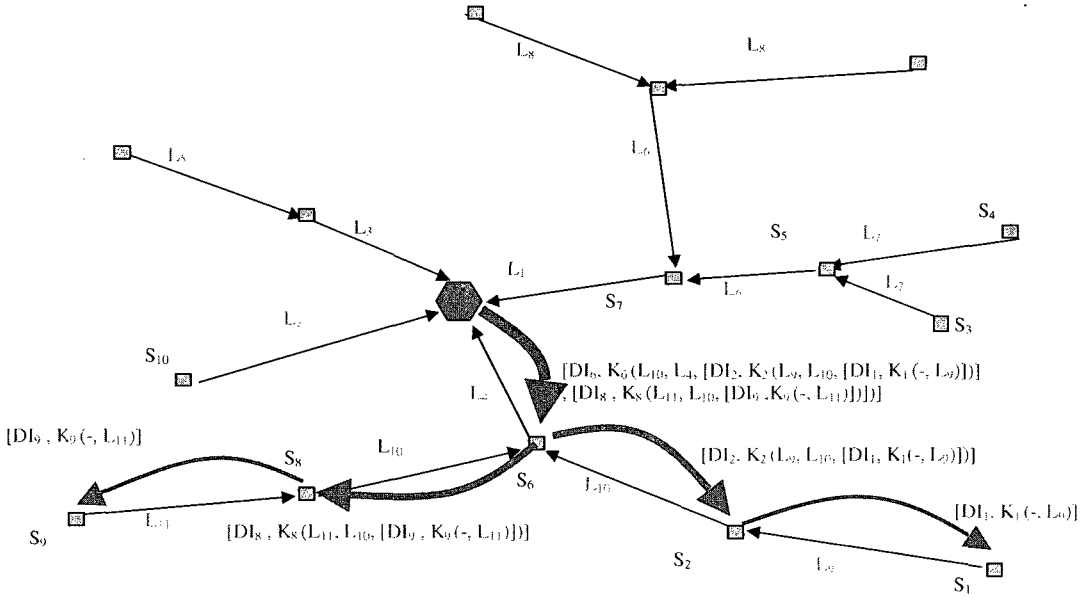


Figure IV-8 Broadcast of a path_setup message

One possible algorithm for assembling the path_setup message for a main branch is the following: the sink starts at any leaf sensor and includes its outgoing label in the innermost layer of an onion. Then, it moves upstream and for each sensor adds a layer to the onion containing its incoming and outgoing labels. If it encounters a "branching sensor" (a sensor from where multiple branches start) it marks the onion so far built as a

sub-onion. It then moves down on each of other branches, and for each one starts building a sub-onion beginning each at a leaf sensor. Once all the necessary sub-onions have been built, it puts them in a tandem and moves upstream from the branching sensor. This process continues until all the sensors in that main branch have been included in the onion. Please note that the sink applies this algorithm offline, to the routing information that it has collected. When all of the onions constructed in this way are broadcasted in the network, each node knows its incoming and outgoing labels.

The following calculations show why bundled messages generate less control overhead compared to individual setup messages. Consider one branch of the shortest path tree containing n nodes excluding the sink. This branch has n hops. In part 6, we will explain that for the purpose of sink-anonymity, the length of all packets on all links must be equal. Let's denote this length with l. Therefore, the total overhead of individual path-setup messages for the nodes of this branch would be $(1+2+ \ldots + n)\, l = n(n+1)l/2$. On the other hand, bundled messages would create an overhead of nl which is smaller as long as $n > 1$. In fact, the saving factor is $(n+1)/2$. However, the savings are normally more pronounced due to branch splitting.

Assignment of Unique Labels

As Figure IV-9 shows, when a node (B) broadcasts a packet, all nodes in its one-hop neighborhood receive that packet. Therefore, some of the nodes in the 2-hop neighborhood of the recipient (A) get the packet. Thus, in order to avoid recipient ambiguity in a local broadcast, the incoming label of a node must be unique at least in its 2-hop neighborhood. In the label assignment phase, the sink must execute an algorithm

to ensure that each node is allocated a unique incoming label. Below, we provide a heuristic algorithm for this task.

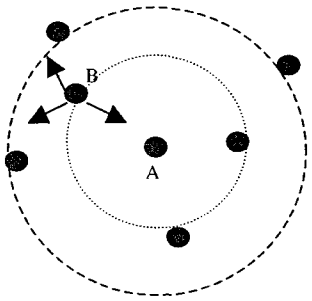


Figure IV-9 A local broadcast

Denote the set of 2-hop neighbors of node X including itself with $N^2(X)$ and the yet unused labels in its 2-hop neighborhood with $L(X)$. At the start of algorithm IV-2, the latter set is the same for all nodes and contains all the available labels. At every step of the algorithm, the sink chooses a node (randomly or systematically) and assigns to it one of the labels still available in its label set. Then, it removes this label from the set of labels of that node and all its neighbors in a 2-hop area. One run of the algorithm ends when all the nodes have been allocated an incoming label.

At the beginning of the next run, the set of unused labels for each node is re-initialized to contain all the available labels. In the following pseudocode, L is the set of available labels and l_x is the incoming label assigned to node x. The rest of the notations are taken from Table IV-1.

Algorithm IV-2 Unique label distribution

Procedure *label_distribution;*

(Input: N, $\{N^1(x) \mid \forall x \in N\}$, L)

(Output: $\{L(x) \mid \forall x \in N\}$)

Begin

For all x **in** N

$L(x) = L;$

For all x **in** N

$l_x = l;$ // $l \in L(x)$

For all y **in** $N^2(x)$

$L(y) = L(y) - \{l\};$

End;

In order to reduce the communication overhead in the downlink we would like the labels to be as short as possible. However, short labels provide a small label space because the number of available labels to use is smaller. Because the DI labels are assigned before deployment and because the sensors are normally distributed randomly, these labels must be globally unique. Therefore, for a network of N nodes, we need a label pool of the size at least N. This means that each DI label must be at least $\log_2(N)$ bits long in order to guarantee that there will be no label collisions. In the uplink direction, labels only have local significance because they are assigned after a round of topology discovery. Therefore, they only need to be unique in a limited hearing area. In other words, they can be spatially reused as long as they do not collide in their locality.

Assume that on average there are n nodes in a 1-hop neighborhood. This value reflects the deployment density of sensors, depends on the application and is considered

an input to our protocol. From there, we find that there are on average *4n* nodes in a 2-hop neighborhood. Therefore, A's upstream incoming label must be unique at least in a set of *4n* nodes. In order to have a label space of this size, a label must be represented by at least $\lceil \log_2(4n) \rceil$ bits. Of course, in order to account for cases where the number of nodes in a neighborhood is actually more than the average, we need more available labels. Adding one bit increases the size of the upstream label space by a factor of two. Thus, we will assume that a label has a length of $l = \lceil \log_2(4n) \rceil + 1$. Longer labels provide better in-ambiguity because they allow a label to be reused less frequently. In other words, instead of making labels unique in a 2-hop neighborhood, we can afford to make them unique in a larger area. They also provide more security because they present a larger anonymity set to an adversary trying to uncover the incoming label of a node. However, it is obvious that longer labels consume more bandwidth and transmission power as well as more processing power when encrypted. In practice, our labels will not be an excessive source of power consumption. A label length of 16 bits (a reasonable assumption) would provide a label space of size 2^{16}, a pool big enough to address almost 64000 nodes. To address hundreds of nodes, a label length of 10 bits is enough. We will explain later in the data transmission part that each data packet carries only one label.

As will be explained in part 6 of this section, in order to thwart traffic analysis attacks, we need the two kinds of labels to be the same length. Therefore, in case $\log_2(N) > \lceil \log_2(4n) \rceil + 1$ we have two options; either make the uplink labels longer or risk the occasional collisions between DI labels. It is unlikely that this inequality is in the reverse order because usually N >> n. If we decide to keep DI labels as short as the upstream labels, we have to reuse them, which means, sometimes two or more adjacent

nodes may share the same DI label. However, this is not a big problem because it only causes one of them to unsuccessfully attempt to decrypt a downstream control packet intended for the other one.

Part 5: Uplink Data Transmission

When a sensor has a packet, it first encrypts it for the sink. Then, it appends its outgoing label to the packet. The upstream neighbor, with an incoming label matching the packet label, accepts the packet and switches its label to its own outgoing label. At this point, it can rebroadcast the packet. However, to prevent a global eavesdropper that applies content analysis to the payload in order to trace the packet back to the source and/or destination, we make the payload to appear different at each hop. To do this, the forwarding sensor encrypts the already encrypted payload for the sink. Every sensor in the path repeats the encryption process, effectively creating an onion. Therefore, on each hop, the packet contains an unencrypted label and a payload that has been encrypted several times. Upon receiving the packet, the sink performs recursive decryptions to recover the original data. Please note that in order to make all data packets in the network of the same length L, the source nodes may have to sometimes apply padding to the original data. Figure IV-10 depicts transmission of a data packet from S_1 to the sink.

Upon receiving the packet, the sink performs recursive decryption on the onion until it recovers the original data. Although the sink may have many shared secret keys, at each iteration, it only needs to try the keys that it shares with the nodes in a 1-hop neighborhood before it can successfully decrypt one layer of the onion. This way, we have used onion cryptography instead of per-hop encryption to thwart content analysis attacks.

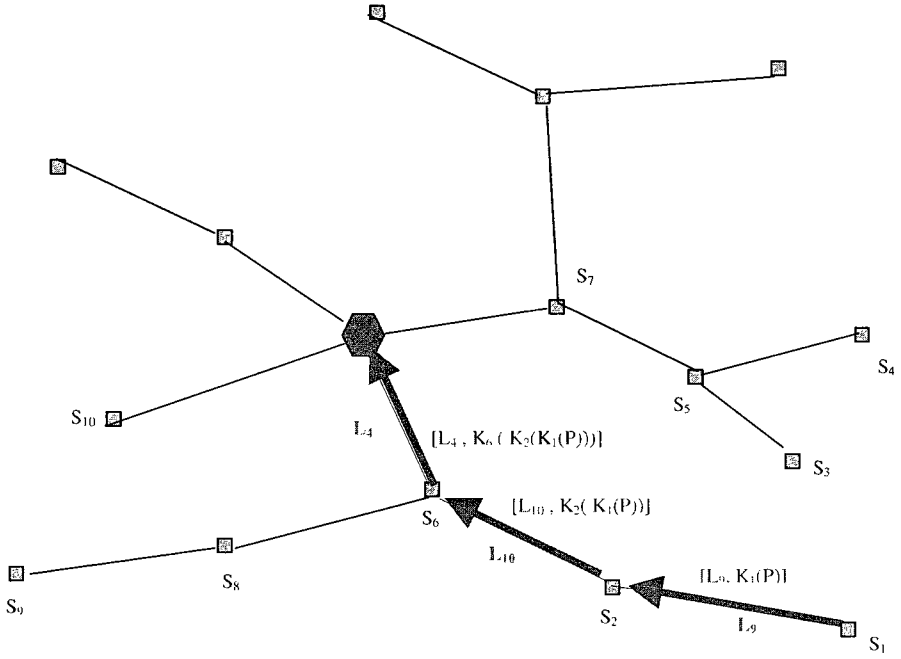


Figure IV-10 Uplink data transmission

Part 6: Downlink Data Transmission

Downlink communications (from the sink to the sensors) is also based on layered cryptography and label switching. We explained before how the sink sends path_setup messages to all the sensors that belong to the same main branch in the shortest-paths tree. Other messages, such as acknowledgments and other control commands are sent in a

similar manner. Remember that each node is initialized with a DI (Downstream Incoming) label. When the sink wishes to transmit a packet, it labels the outermost layer of an onion with the DI of one of its neighbors that is supposed to be the next hop for that packet according to the latest routing information available to the sink. However, inside that layer, which is encrypted for that neighbor, it precedes each sub-onion with the DI label of the next hop on the route. The path setup process illustrated in Figure IV-8 is an example of downlink data transmission. For enhanced security, DI labels may be changed periodically.

Sink Anonymity in Downlink Communications

In order to prevent an eavesdropper from acquiring valuable information regarding the location of the sink, path_setup messages and other control packets sent by the sink in the downlink must be indistinguishable from data packets transmitted by sensors. Data packets consist of a label and an encrypted payload that is re-encrypted on every hop. Control packets look exactly like data packets since they have a label (DI) and an encrypted part that looks different on each hop because of layered cryptography. Moreover, these packets are only transmitted within a short setup phase, which gives the adversary little chance to find the sink while in other protocols any compromised node gives a clue about the sink's location. An issue that must be noted here is that the length of the control packet and the data packets must be a constant and equal on all the links in order to stop content analysis attacks. In the uplink, the length of the data packets does not change because they are only re-encrypted and the source node can apply the necessary padding at the origin to makes them the same length. However, in the

downlink, control packets get shorter after each repetition because each forwarding node removes its labels from the peeled layer. Therefore, the forwarding node must add an equal amount of random padding to the end of the packet, similar to how it is done in the onion routing protocol.

Upon hearing a packet, a sensor executes Algorithm IV-3.

Algorithm IV-3 Packet forwarding decision on a node

Procedure *process_packet;*
(Input: Packet P)
Begin
If *($label_P$ == my_DI)* **Then** *//it is a downlink packet and I am the immediate recipient.*
    1: *decrypt and process the payload.*
    2: *strip off my DI label and rebroadcast sub-onion(s).*
Else If *($label_P$ exists in my routing table)* **Then** *//it is an uplink (data) packet*
    1: *swap label* *//am the immediate recipient.*
    2: *re- encrypt payload*
    3: *forward packet.*
Else
    1: *discard packet.*
End;

IV. 8 SECURITY ANALYSIS

By using onion cryptography and by making all data packets the same length on all hops, we have made sure that data packets on consecutive hops are uncorrelatable[11]. Therefore, assuming there is sufficient traffic in a neighborhood, every transmission in that area may equally likely correspond to an original data packet from a source node or to the re-transmission of a packet from a downstream node. In fact, this is the main reason for source anonymity and location privacy in DCARPS. However, in reality, this assumption may not always be true. In other words, assuming we do not use cover traffic to confuse adversaries, the volume of traffic in a particular neighborhood may be less than a minimum when a source node broadcasts a packet. In that case, an adversary has an opportunity to identify and locate the source of that packet. In particular, if a node transmits a packet while there has been no previous transmission in its 1-hop neighborhood, it can be easily concluded that the sender is in fact the source node.

This technique can be used by an attacker to uncover the incoming labels of different nodes. The outgoing label of a node is not a secret and can be seen in every packet transmitted by that node. Given enough time, depending on the traffic model, a patient global adversary may be able to discover the incoming labels of all the nodes and eventually reveal the links between them. The following rules are interesting to the adversary and can be used as attack strategies:

[11] Anti-timing-analysis techniques are sometimes used to obfuscate the timing relationship between transmissions on consecutive links. A simple re-ordering of a few buffered packets before re-transmission is one of these techniques. However, even without these methods, the timing of transmissions in an area are normally randomized to a degree, due to the random nature of channel access in MAC protocols.

Rule #1. *If node X sends a packet when there has been no previous transmissions from other nodes in its 1-hop neighborhood for a certain amount of time, X is the source of that packet.*

If the incoming label of X is already known to the adversary, he need not consider previous packets bearing labels other than X's incoming label in this process because they would not be repeated by X anyway.

Rule #2. *If X and Y are the only nodes transmitting within a certain amount of time, and the first transmission is done by node X, then Y is the upstream node from X and the incoming label of Y is the same as the outgoing label broadcasted by X.*

This is assuming that all packets are repeated and no packet is dropped. However, in reality some packets are lost which adds to the confusion of the attacker. Moreover, the sink does not repeat most of the packets that it receives. Therefore, some of the packets sent to it may be wrongly associated by the attacker with other transmissions in its neighborhood.

Rule #3. *If all the neighbors of a node X have been linked to other nodes as their parents using Rule #2, then X is a leaf node because no node can have two parents.*

A patient adversary may use rules 2 and 3 and the process of elimination to gradually uncover the parent-child relationships between the nodes. For every discovered

incoming label, the size of the anonymity set (set of possible values) for the unknown labels in the 2-hop neighborhood of the corresponding node is reduced by 1[12]. On the other hand, when a node Y, a neighbor of node X, transmits a packet bearing a label other than the incoming label of X, it can be concluded that X is not the parent of Y. Note that in DCARPS, each node has one and only one parent but it can have several children.

Once the adversary acquires complete knowledge of incoming labels, if a node X transmits a packet while there has been no packet previously in its 1-hop area whose label matches its incoming label, the adversary can be certain that X is the source node. Moreover, a leaf node is the source of any packet that it transmits because it does not have any downstream node. However, these risks can be alleviated by periodically (this period depends on several factors including the traffic model and the strength of the adversary) changing incoming labels of nodes. Besides, X may generate its own packets when it has packets to relay as well. In that case, the anonymity set for the source of any packet transmitted by X includes X and all the nodes downstream from it. In the rest of this analysis, we will assume that the adversary relies on Rule #2 to identify links between nodes. The success of the adversary depends on the practicality of this method.

Recognizing the parental relationship between nodes is further complicated for the adversary by the fact that data packets (in the uplink) are indistinguishable from control packets (in the downlink). In other words, each node (except leaf nodes) has one incoming label in each direction and can be sending a packet in either direction at any time. Therefore, while the attacker is trying to correlate transmissions in one direction it

[12] Remember that in part 4 of the DCARPS protocol description we concluded that a label must be unique in a 2-hop neighborhood and that the size of the label space must be at least 4n where n is the average number of neighbors in a 1-hop area.

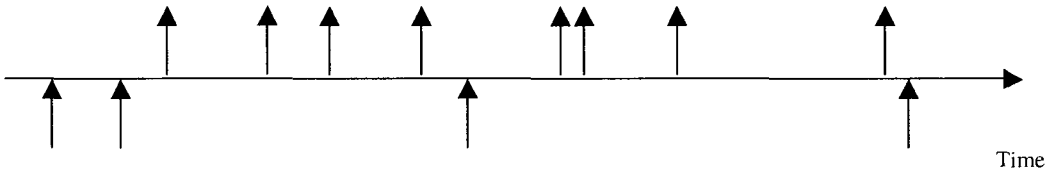
has some difficulty in identifying the direction of a packet. For example, assume that the adversary has found one of the two incoming labels of node X. Now, if a packet is transmitted in the 1-hop neighborhood of X with a different label, the adversary cannot safely rule out X as the possible recipient because this packet may be intended for X in the opposite direction.

Due to its unique role in the network, the sink is more vulnerable to traffic analysis attacks. The fact that the sink uses a variety of labels to send downlink packets could be a hint to an attacker. But, all the branching nodes in the network demonstrate a similar behavior. Another property of the sink is that it does not repeat data packets (this feature (i.e. fake retransmissions) may be added to the original design for more security).

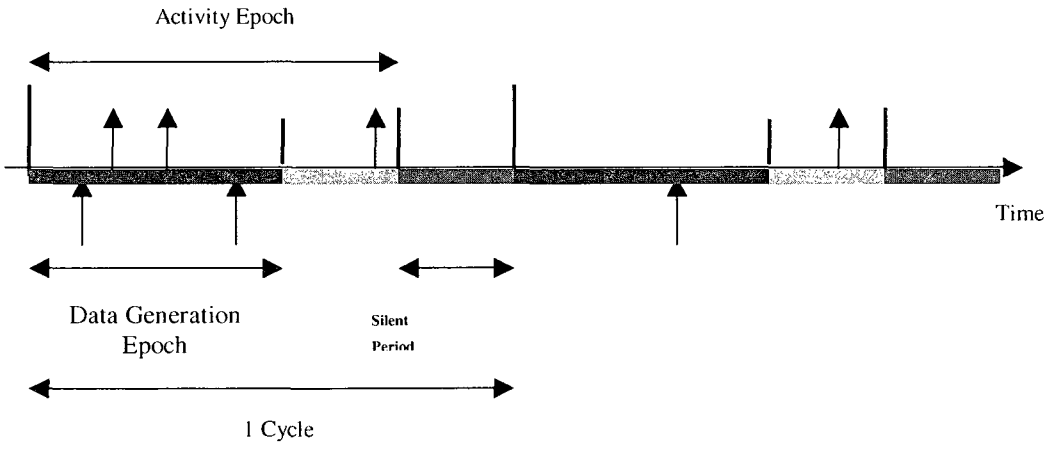
The special role of leaf nodes can also help an attacker in telling apart the uplink and the downlink labels. A leaf node has one DI label and no incoming label in the uplink direction. Moreover, it does not repeat downlink messages. Therefore, when the parent of a leaf node sends a path_setup message to it, the fact that this packet is not repeated (known from using rule #2) is a hint to the attacker that it must have been a downlink packet. Also, the label that the leaf node uses to send packets is an uplink label. From there, the directions of both incoming labels of all the upstream nodes from that leaf node are known.

Rule #4. *If a node X transmits a packet, which is followed by no transmission in X's 1-hop neighborhood for a certain amount of time, then X is either a neighbor of the sink or the parent of a leaf node.*

As can be seen, the success of the adversary depends on several unlikely conditions including, no packet loss, frequent violation of the high traffic volume requirement, unchanging label assignments and limited use of fake packets and anti-traffic analysis techniques. A stochastic analysis for the probability of success of the adversary and the length of the safety period (the amount of time that an adversary needs to reach its goal which may be finding all the incoming labels or finding the sink) is desirable.



a. Random Traffic Model in Sensor Networks



b. Cyclic Traffic Model in Sensor Networks

Figure IV-11 Random and Cyclic Traffic Models

Sensor networks can be categorized in terms of their traffic models. From that point of view, there are two main groups of sensor networks, one in which data may be generated and forwarded at any time (random model) and the other where traffic exists only within activity epochs (cyclic model). The duration of a cycle is called the wake-up period. In Figure IV-11, original packet arrivals are represented by arrows below the time axis while forwarded packets are shown by arrows above the time axis. The two diagrams in this figure show example traffic patterns in a 1-hop neighborhood. In order to simplify analysis, we assume that all packets generated within a data generation epoch are delivered to the sink within the overlapping activity epoch. In other words, all nodes start a new activity epoch with empty buffers. Obviously, in practice this may not be the case, which makes the work of the adversary more difficult because it would have to keep track of packets from one activity epoch to the next. In the second portion of the activity epoch, nodes continue relaying packets but do not generate new packets.

As far as our analysis is concerned, we believe that these two traffic models can be treated the same way because the random model is in fact a special case of the cyclic model where the duration of the silent period is zero and the durations of the data generation epoch and the activity epoch are equal.

Assume that there are N active nodes in the network and an average of n nodes in a 1-hop area. Let's divide the activity epoch into time slots of length w seconds each and assume that a node buffers all the packets received within one time slot to forward them (perhaps after shuffling) within the next time slot. Therefore, a packet is repeated after a delay of at least zero and at most $2w$ seconds. Each transmission follows a random backoff period that is applied in order to prevent collisions. Therefore, the order of

packets arriving in a 1-hop neighborhood is de-correlated from the packet departures from that neighborhood.

Let's assume that the random variable K, the number of packets transmitted (original and repeated) within one time slot in a 1-hop neighborhood has a Poisson distribution with an average of λ packets per slot and a probability density function P_k. In order to evaluate the security of our protocol, we must do the following:

a. Calculate the probability of the event used in Rule #1, in order to find the probability that the source of a packet can be revealed.
b. Calculate the probability of the event used in Rule #4, in order to find the probability that the sink can be located. Other techniques such as Rule #3 and the fact that the sink generates more traffic and uses multiple outgoing labels may be used to differentiate between the sink and the leaf nodes.
c. Calculate the probability of the event used in Rule #2, in order to find the probability that X's parent can be identified with only one transmission form X.
d. Calculate the *safety period* of the protocol, defined as the time needed to find all the uplink incoming labels. Please note that this definition of the safety period represents, in a sense, a worst-case scenario because the adversary needs as well to find the incoming labels in the downlink direction. However, because a node can have any number of children in the downlink, our analysis would have been too complicated. Therefore, we are assuming that in the time interval when the adversary attempts to perform its investigation, there are only uplink transmissions, which would be a big advantage for him. Please note that even

then, the attacker can at most identify the sink. The source of a particular data packet cannot be identified even with full knowledge of the routing tree because forwarded and original packets are indistinguishable.

Problem a

Assume that there are some transmissions in a time slot. Find the probability that there are no transmissions in the previous time slot. In that case, all the transmitted packets would be original and belong to their observed senders.

Solution

As we assumed, the number of packet transmissions in a time slot has the following probability density function:

$$P_k = \frac{\lambda^k}{k!} e^{-\lambda} \qquad (1)$$

Hence, the probability that the sources of these packets can be detected using Rule #1 is:

$$P_0 = e^{-\lambda} \qquad (2)$$

λ (the aggregate packet arrival rate in a 1-hop neighborhood) depends on the nature of the application and indicates how high or how low this probability is. The higher this value the better the security is. Conversely, for a desirable level of source location privacy based on Rule#1, the minimum required amount of traffic may be calculated using (2). A special case of this problem is when only one node X transmits in a time slot. In that case, all packets transmitted by that node in that time slot are original and belong

to X if and only if no other node than X transmitted in the previous time slot. Assume that a node transmits in a time slot with probability P_T. Then, the number of transmitting nodes in a time slot is a binomial random variable. Therefore, the probability that none of the neighbors of X transmits in the first time slot is:

$$P_0 = \binom{n-1}{0} P_T^0 (1-P_T)^{n-1-0} = (1-P_T)^{n-1} \qquad (3)$$

Obviously, this probability depends on the transmission rate of the sensors (a characteristic of the application) as well as the sensor deployment density. The higher these parameters the better the security.

Problem b

Find the probability that a node X can be identified as a neighbor of the sink or the parent of a leaf node.

Solution

This probability is the probability of the joint event that X transmits in a time slot and that none of its neighbors transmit in the next time slot. The probability that X transmits in the first time slot is P_T. The probability that none of its neighbors transmit in the next time slot is the same as P_0 calculated in (3) because as far as the adversary knows these events are independent of each other. Therefore, the probability of the joint event is:

$$P_T (1-P_T)^{n-1} \qquad (4)$$

Problem c

Find the probability that Y, the parent of a node X can be identified with only one transmission by X.

Solution

The probability that only X transmits in a time slot is:

$$P_X = \binom{n}{1} P_T^1 (1-P_T)^{n-1} = nP_T(1-P_T)^{n-1} \qquad (5)$$

The probability that only one of the neighbors of X (in this case Y) transmits in the next time slot is:

$$P_Y = \binom{n-1}{1} P_T^1 (1-P_T)^{n-1-1} = (n-1)P_T(1-P_T)^{n-2} \qquad (6)$$

Again, due to the independence of the two events, the probability of interest is:

$$P_X.P_Y = n(n-1)P_T^2(1-P_T)^{2n-3} \qquad (7)$$

Problem d

Find the safety period.

Solution

From a Statistical point of view, the safety period is the time that the adversary needs to find the parent of one single arbitrary node in both directions. Because, presumably a global adversary tries to find all the parent-child relationships concurrently and it progresses at the same rate with regards to all the nodes if the traffic pattern is uniform

across the entire network. At the beginning, as far as the adversary knows, all of the neighbors of an arbitrary node X are equally likely to be its parent i.e. the anonymity set for X's parent is the set of all its 1-hop neighbors. With every transmission from X in one time slot, this anonymity set shrinks to be its intersection with the set of neighbors that transmit in the following time slot. This is true because, X's parent must be in the anonymity set and it must also be among the nodes that make transmissions in the following time slot since we have assumed all packets are repeated until they reach the sink.[13] In other words, if $AS_X^{\ i}$ denotes the anonymity set for X's parent in the i^{th} time slot and $TS_X^{\ i}$ is the set of those neighbors of X that make transmissions in the i^{th} time slot, and X transmits at least one packet in the $(i-1)^{th}$ time slot, then:

$$AS_X^{\ i} = AS_X^{\ i-1} \cap TS_X^{\ i} \qquad (8)$$

In a particular time slot, members of TS_X can be any of n-1 neighbors of X while members of AS_X can only be some of them that are still likely to be the parent of X.

The safety period is the time that the adversary needs to reduce the above-mentioned anonymity set to a one-member set containing only the parent of X. In terms of information theory, the adversary starts with full uncertainty about the parent of X (highest entropy) and gradually reduces its uncertainty. At the end of the safety period, the adversary has an uncertainty of zero regarding the parent of X and other nodes for that matter. The rate at which this uncertainty decreases depends on the transmission patterns of X and its 1-hop neighbors. Indeed, the anonymity set for X's parent in each time slot is a random variable with memory whose outcome depends not only on another

[13] The adversary may consider auxiliary rules as well in order to reduce the size of this set faster. One such rule is the fact that the parent of node X must transmit at least as many packets in one particular time slot as X does in the previous time slot. As another example, if one of the neighbors of X is revealed to be its child, then it cannot be its parent.

random variable (TS_X) but also on its own value in the previous time slot. Let's denote this random variable with Y. The sample space of Y in the i^{th} time slot is the set of all the subsets of AS_X in the previous time slot except the null set because X must have one and only one parent.

$$S_Y{}^i = \left\{ y_j \mid y_j \subset AS_X{}^{i-1}, y_j \neq \Phi \right\} \tag{9}$$

On the other hand, the sample space for TS_X in all time slots is the set of all possible subsets of N_X, the set of neighbors of X.

$$\begin{aligned} N_X &= \{n_1, n_2, ..., n_{n-1}\} \\ S_{TS_X}{}^i &= \left\{ t_j \mid t_j \subset N_X, t_j \neq \Phi \right\} \end{aligned} \tag{10}$$

S_{TS_X} has 2^{n-1}-1 members (different combinations of neighbors of X). The entropy of Y in a time slot is a measure of uncertainty of the adversary in that time slot regarding the parent of X. If $AS_X{}^i$ contains r_i neighbors, then $S_Y{}^{i+1}$ will have 2^{ri}-1 members, likelihood of each will depend on the likelihood of members of S_{TS_X}. $P(y_j)$ can be calculated by dividing the number of members of S_{TS_X} whose intersection with $AS_X{}^i$ is y_j by 2^{n-1}-1. For any y_j, there are 2^{n-1-ri} members of S_{TS_X} whose intersection with $AS_X{}^i$ is y_j. Thus:

$$P(y_j) = \frac{2^{n-1-r_i}}{2^{n-1}-1} \tag{11}$$

If n >> 1, then for all y_j's, $P(y_j) = 2^{-r_i}$. The entropy of Y in the $(i+1)^{th}$ time slot is:

$$H(Y) = -\sum_{j=1}^{2^{r_i}-1} P(y_j)\log P(y_j) = r_i(1-2^{-r_i}) \quad (12)$$

The safety period T_{safety} is actually the number of time slots that it takes for H(Y) to merge to zero. r_i is initially equal to n-1 (when the adversary starts its efforts) and then decreases or remains the same with every transmission by X. In order to calculate the average number of time slots that we need to reduce H(Y) to zero, we have to first calculate the average reduction in the size of the anonymity set from one time slot to the next. If there are r^i nodes in the anonymity set in the i^{th} time slot, then the anonymity set in the $(i+1)^{th}$ time slot has a size of $1 \le r_{i+1} \le r_i$. The probability that $r_{i+1}=m$ is:

$$P(r_{i+1} = m) = \binom{m}{r_i} P(y_j) = \binom{m}{r_i} 2^{-r_i} \quad (13)$$

The expected value of r_{i+1} is:

$$E(r_{i+1}) = \sum_{m=1}^{r_i} m \binom{m}{r_i} 2^{-r_i} \quad (14)$$

And the expected value of reduction in r_i is:

$$E(r_i - r_{i+1}) = r_i - \sum_{m=1}^{r_i} m \binom{m}{r_i} 2^{-r_i} \quad (15)$$

Using the previous two equations and considering $r_1 = n-1$, the expected number of transmissions from X to find its parent (denoted by γ) can be calculated from the following equation:

$$\sum_{i=1}^{\gamma} E(r_i - r_{i+1}) = n-2 \quad (16)$$

However, a node does not transmit in all time slots. If a node transmits in a time slot with probability P_T, then the expected value of the safety period in terms of time slots is:

$$T_{safety} = \frac{\gamma}{P_T} \qquad (17)$$

We have to mention again that this is a lower bound for the safety period.

IV.8.1 Simulations

In this section, we provide the results of our OPNET simulations which were used to evaluate the level of security provided by DCARPS. This simulation study is intended to complement the stochastic approach of the previous section.

The principle premise of almost all anonymous routing protocols is confusing the adversary using randomized routing, which loosely defined means selecting the route of the packet either genuinely randomly or at least making it look random from the perspective of an eavesdropper. Therefore, the effectiveness of these protocols depends on the level of randomness that they generate for the adversary. Hence, we propose that their performance levels may be evaluated using *entropy* as a measure of uncertainty of the adversary.

Entropy is commonly used as a measure of uncertainty associated with a random variable. The entropy of a discrete random variable X with a sample space of $\{x_1, \ldots, x_n\}$ is defined as $H(X) = \sum_{i=1}^{n} - p(x_i).\log p(x_i)$. In the case of a randomized routing protocol such as DCARPS, it can be used to measure the adversary's uncertainty in identifying

different aspects of a transmission e.g. the source and the destination of a packet. Assuming the routing is identity-free (packets do not reveal the IDs of the end nodes) as is in the case of DCARPS, communication anonymity can be violated only if both the source and the destination are located and identified.

For both source and destination privacy, the strength of DCARPS depends on the inability of the eavesdropper to correlate the consecutive packet transmissions in a 1-hop area i.e. identify the immediate sender and the immediate receiver of a packet in each local transmission.

Methodology

Source Privacy: Run simulations for a large number of packets generated randomly at different source nodes. Use DCARPS to forward packets until they reach the destination. After each packet transmission by a node (new or relayed packet), consider those of its neighbors who transmitted within the last w-second time interval to be the anonymity set for that packet. These nodes represent the set of nodes that the adversary considers as possible immediate senders of that packet. Remember that due to the contention-based wireless transmission medium and per-node queuing and processing delays, even without the use of packet shuffling techniques, two packets arrived at two neighboring nodes may not be repeated by them in the same timing order. In other words, the packet arriving earlier at one of the nodes may experience more delay and finally be relayed later than the other packet by the other node. Thus, the adversary cannot assume exact timing information and must consider a minimum observation period for each packet within

which any transmitting neighbor might be the sender of the packet in question. We refer to this observation window as the *entropy period.*

If the size of the anonymity set mentioned above over the length of the entropy period for a certain packet is s and all of these nodes are equally likely to have been the previous sender of the packet then the uncertainty of the eavesdropper for that packet can be calculated as $-\sum_{i=1}^{s}\frac{1}{s}\log(\frac{1}{s}) = \log(s)$. We consider the base of the logarithm to be 10. At the end of the simulation, the overall entropy of the node is considered to be the average of its per-packet entropies. If the node transmits a total of m packets, its overall entropy will be $\frac{1}{m}\log\prod_{i=1}^{m}s_i$. We also use the mean of individual node entropies as the overall protocol entropy.

Destination privacy is calculated in a similar manner. When a node transmits a packet, from the point of view of the eavesdropper, any one of its neighbors that happen to transmit within the next w-second window may be the next hop and thus a member of the per-packet next-hop anonymity set. The rest of calculations are similar to the source privacy case.

Simulation Parameters

We experimented with networks of size 20 and 50 nodes with various topologies obtained from uniformly distributed random node placements over a 100 x 100 area. We investigated the variations in the source and destination entropy as impacted by different parameters including node transmission range, node density, traffic volume, node packet forwarding delay and entropy period. In each case, we ran the simulation long enough for

the results to reach a plateau which meant a sufficient number of packets were handled. Below, we will discuss the impact of each one of these factors.

Traffic Volume: This is one of the most important parameters affecting the performance of DCARPS. It depends on the nature of the application making use of the sensor network and in particular the sensor data generation rates. Considering the very wide range of applications for sensor networks, the value of this parameter could vary significantly. As our simulation results will show, the entropy of the protocol increases with this parameter. However we needed to experiment with reasonable values for this parameter which would not be impractical in real life situations. In our survey of existing wireless sensor networks systems, we found some useful information in this regard. For example, it is normal for ECG (Electrical CardioGram) signals to produce data at a rate of 3.2 kbps. A wireless sensor network of such ECG machines could be formed at a hospital, a project which is incidentally a good example of the need for user anonymity and location privacy. The famous IEEE 802.15.4 (Zigbee) standard allows a maximum packet size of 127 bytes which to the best of our knowledge is currently the largest packet size for sensor networks (resulting in minimum traffic volume in terms of packets). However, a big part of each packet in this standard is reserved for the headers. If we generously allow 100 bytes of user data per packet, an ECG machine would need to produce 4 packets per second. In some industrial applications, data submission rates of up to 60 samples per second can be found and audiovisual sensors generate a high volume of data too. Moreover, concurrent applications in sensor networks increase the traffic volume by combining the packets generated by different applications in the same

network. In our simulations, we experimented with a Poisson distributed packet generation process at each sensor at rates of between 0.5 and 20 packets per second.

Node Delay: We considered an exponentially distributed node delay comprising of queuing delay, channel access delay and perhaps user induced deliberate delays to counter timing analysis attacks. Therefore, this parameter depends on the transceiver bitrate and network load as well as any anti-traffic analysis technique that may be used. In order to determine realistic values for this parameter, again consider 127-byte packets and a mean service rate of 5080 bps at a node. This results in a mean delay of (127 x 8)/5080 = 0.2 seconds per packet. The available bitrates of commercial sensor devices vary between 9.6 kbps for the first generation and 250 kbps for Zigbee-compatible MICAz motes. Therefore, even the slowest radios leave some room for the other two kinds of delays. We experimented with 3 different values for this parameter namely 0.2, 0.4 and 0.6 seconds.

Entropy Period: This parameter has a critical role in the uncertainty of the adversary. Proper selection of its value in relation with the node delay increases the adversary's success ratio. Choosing a small value for this parameter results in a smaller uncertainty for the adversary but it may also result in too many false positives. In other words, the anonymity set for each packet would be smaller which is desirable for an eavesdropper but it may not contain the actual sender or the receiver of the packet leading him to make a wrong determination. On the other hand, if this period is too long compared with the node delay, the anonymity set would be more likely to contain the neighbor of interest

but it would also have more members resulting in a higher uncertainty. In our simulations, we experimented with values of 0.1, 0.2, 0.4 and 0.6 seconds for this parameter while maintaining the ratio of entropy_period / node_delay at one of the following values: 0.5, 1, 2 or 3.

Transmission Range (Tx): This parameter determines the size of a 1-hop neighborhood and hence influences the average number of neighbors of a node. Therefore, the size of the anonymity set for each packet grows with this parameter resulting in higher entropy levels. We experimented with values between 25 and 55. For smaller values, the network is often partitioned while for larger values it largely resembles a single hop network instead of multihop for which routing is trivial.

Simulation Results

We provide our simulation results for the overall source entropy of DCARPS. The results for the destination entropy were almost identical. Figure IV-12.a shows the overall source entropy vs. packet generation rate per second when node-delay and entropy period are each 0.2 second. In this and all other graphs in this section, each point represents the average of 5 simulation runs with different network topologies. As can be expected, the entropy increases with network load. However, beyond almost 10 pkts/sec, the entropy does not seem to improve by the packet generation rate. The reason is that at that point almost all of the neighbors of a node are transmitting at least one packet within the entropy window for a particular packet and are thus already included in the anonymity set

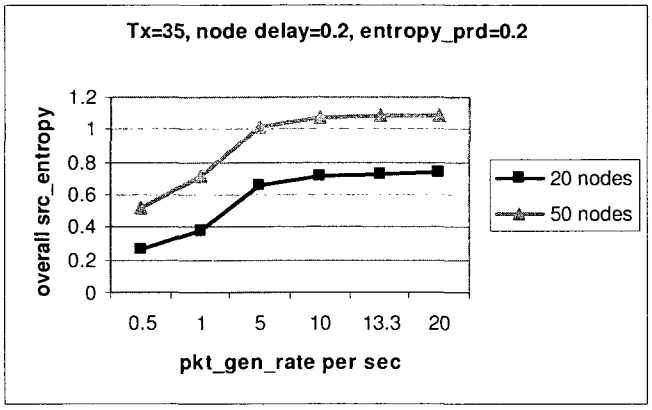
for that packet. Therefore, increasing the traffic volume does not increase entropy. This effect can also be seen in figure IV-12.b, explained later.

Also, for a fixed packet generation rate, increasing the number of nodes from 20 to 50 improves entropy because the traffic volume increases. The entropy has a maximum of almost 0.75 for 20 nodes and 1.1 for 50 nodes.

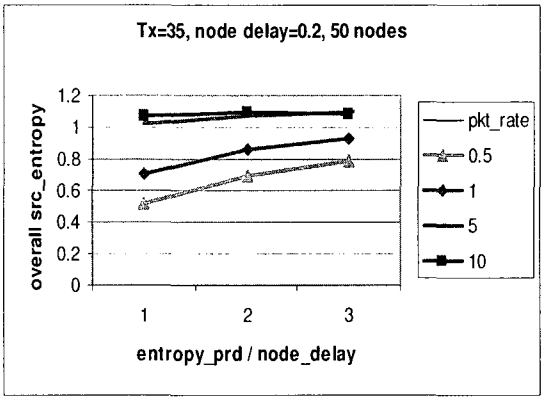
To put things in perspective, please note that entropy of 1 represents the uncertainty associated with a random variable with a sample space of size 10 with equi-probable members. In other words, it is as though the eavesdropper is uncertain which one of 10 nodes is the sender of each packet. On the other hand, entropy of zero represents a network that does not use randomized routing and the eavesdropper can identify the source and the destination of each packet with certainty.

Figure IV-12.b shows how entropy grows with the entropy_period / node_delay ratio for 50 nodes and for different traffic volumes when the node delay is fixed at 0.2 seconds. On the other hand, figure IV-12.c shows that for the same value for this ratio, the entropy increases when the node delay is increased. The reason for this behavior is that as delay increases the entropy period must increase so that their ratio will not change. Increasing the entropy period makes the size of the anonymity set bigger because the probability that more neighbors transmit in that time window will be higher.

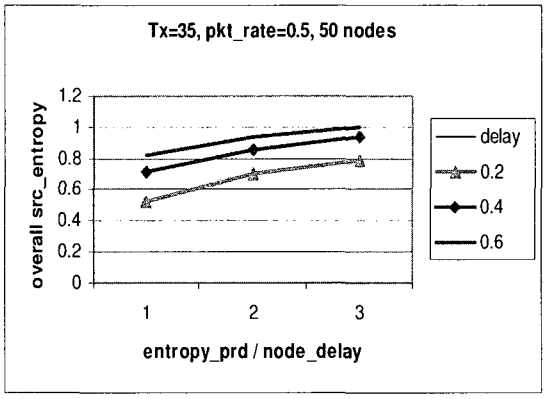
The impacts of transmission range and node deployment density can be seen in figures IV-12.d and IV-12.e, respectively.



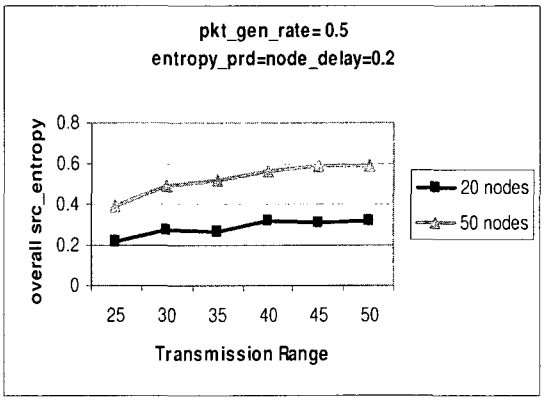
(a)



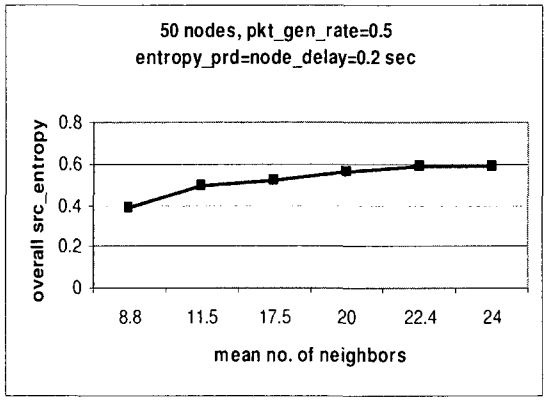
(b)



(c)



(d)



(e)

Figure IV-12 Overall DCARPS source entropy

IV.9 PROTOCOL OVERHEAD

As we explained, DCARPS consists of three phases, topology discovery, path establishment and data transmission. In this section, we discuss the overhead incurred in each part separately.

1- **Topology Discovery**: As mentioned before, we have proposed two possible network topology discovery protocols for this phase. A performance evaluation of one of these protocols was given in part 2 of the protocol description in section IV.7.2. The description and a performance evaluation of the other protocol namely 2hop-clustered-NTDP can be found in chapter VI. Our simulations have shown that the 2hop-clustered-NTDP has a better performance but instead the first protocol is capable of supporting network topology disclosure protection.

2- **Path Establishment**: Due to the fixed size of packets on all links in DCARPS, the overhead of this phase is predictable and on average equivalent of the transmission and reception of one packet per node. Let's take l, the length of a packet, as our unit of overhead. During this phase, each intermediate node sends an onion sublayer to each one of its downstream neighbors. Therefore, each node transmits as many units of overhead as the number of its children on the shortest-paths tree. Leaf nodes do not add to the transmission overhead since they have no children. Because of the spanning tree formation of the shortest-paths tree (each node has only one parent), in a fully connected network, the sum of the numbers of the children of all nodes equals

one less than the number of nodes and this is the total amount of transmission overhead in terms of units of overhead.

Every time a path setup packet is transmitted by a node, all of its neighbors, not just the downstream intended recipient, receive it. However, using the DI labels, the neighbors other than the intended recipient can quickly find out that the packet does not belong to them. Since the labels can be implemented on the link layer (similar to MPLS), these neighbors do not need to receive the entire packet and can abort reception after the DI label has been examined. If we ignore the overhead of a DI label compared to an entire path setup packet, then we only need to consider the reception overhead incurred by the intended downstream recipient. Therefore, the total reception overhead of our path setup phase in terms of units of overhead is also equal to the number of nodes other than the sink. Please note that the overall energy consumption due to reception must include both data reception and data decryption. Transmission of path setup messages by intermediate nodes does not involve any cryptographic operations. Please also note that the transmission and the reception functions of a transceiver consume different amounts of power.

3- **Data Transmission**: The only transmission overhead imposed on the sensors in this phase compared to a regular sensor network is the addition of only one label to each data packet, which is just changed at each intermediate node. However, every intermediate node also re-encrypts the data packet. The cryptographic overhead depends on the size of the packet and the encryption algorithm and beyond the scope of DCARPS. As an example, in [129] the power

consumptions of the commonly used RC5 and IDEA encryption algorithms are compared on several different platforms. For instance, according to this paper, transmission and reception of a 16-byte packet using an MSP processor when encryption and decryption are also considered consumes 4.55uA. However, [130] shows that under certain circumstances, MISTY1 and Rijndael consume much less energy.

IV.10 SUMMARY

In this chapter, we considered the problem of location privacy and anonymity for the sink and the source nodes in an abundant-traffic WSN. We proposed an anonymous routing protocol for this kind of network based on DCR and label switching. This protocol provides source and destination location privacy and communication anonymity due to its identity-free routing property. We allowed the sink to take charge of all route calculations which saved the energy of the sensors as well as eliminated the need for sink-originated beacons that reveal its location. Instead, we proposed and evaluated a new topology discovery protocol that provides the sink with a global view of the network topology.

In the process of route calculations, we used a load balancing algorithm in order to maximize the lifetime of the network. In order to protect against packet tracing attacks, our DCARPS anonymous routing protocol uses layered symmetric cryptography and equi-length packets as well as a similar format for the control and data packets. Stochastic as well as simulation security analyses of this protocol were also provided. In the next chapter, an extension of DCARPS called Probabilistic DCARPS is proposed for scarce-traffic networks.

The use of labels in DCARPS instead of node IDs makes it an identity-free routing protocol. This property makes the protocol more anonymous because the node IDs are not present in the packets.

In order to assess the capability of DCARPS in terms of hiding the identities and the locations of the source and the destination nodes, we proposed a new analytical method based on entropy. In fact, we introduced a metric that measures the uncertainty of an adversary in determining the source or the destination of packets.

For simplicity, at this time, we assumed a network of stationary nodes, but our protocol can also be applied to an environment consisting of mobile sensors and mobile sinks. We also, assumed a single sink but an extension of our protocol to multiple independent or cooperating sinks is straightforward. It is also possible to extend our protocol to a hierarchical architecture (for scalability), in which multiple sinks are organized into layers, each responsible for a subset of sensors ultimately reporting data to one top-layer sink.

CHAPTER V

PROBABILISTIC DCARPS

In this chapter we adapt DCARPS to protect the location privacy and anonymity in wireless sensor networks that handle few data flows, albeit against a more limited attack model compared to DCARPS as will be explained.

As was pointed out in the previous chapter, the basic DCARPS, similar to most anonymity schemes relies on the existence of a sufficient amount of traffic flows (a large anonymity set) to be able to provide a reasonable degree of security. It confuses an adversary by making a big number of packets indistinguishable so that packet tracing attacks can be resisted against. While this condition is intrinsically satisfied in many types of sensor networks such as inventory tracking, habitat monitoring, environment monitoring and some military applications, it may not be the case in some scenarios such as law enforcement, recovery operations and some military situations. In other words, we may have a sensor network with one or few data streams, for example when a reconnaissance unit is sending information to the command centre. Although fake traffic can be generated in the network in order to artificially satisfy the minimum traffic volume requirement, this method is not practical in energy-constrained and bandwidth-limited sensor networks. In this chapter, we extend DCARPS so that it can provide reasonable anonymity and location privacy in such an environment.

In section IV.5, we explained that a scarce-traffic network cannot be protected against a global adversary but with proper route randomization, it can be protected against a local eavesdropper. When fixed-path routing (as opposed to randomized routing) is used, path tracing attacks cannot be stopped because they usually use physical layer means. In other words, regardless of what is done at the network layer, the adversary is guaranteed to arrive at the source if it keeps moving to the immediate sender of any new packet belonging to the same packet stream. In a similar manner, the adversary can be guided towards the sink by listening to uplink transmissions from upstream nodes, moving up one hop per packet. The Phantom routing protocol described in chapter II uses phantom sources to solve this problem. It sends each packet from the real source to a different fake (phantom) source, using a directed walk and then the phantom source sends the packet to the sink, using either flooding or single-path routing. In effect, it starves the attacker from a steady stream of packets that it can backtrack towards the real source. However, this protocol has shortcomings that were discussed before.

As long as the label assignments remain unchanged, DCARPS offers fixed end-to-end paths. In order to extend this protocol to support scarce traffic situations, we have to change this fixed-path routing behavior. In the next section, we will propose a probabilistic packet forwarding approach in order to distribute a single packet stream randomly over multiple paths, effectively denying the adversary access to a steady stream of packets. Specifically, we incorporate some randomization in the packet-forwarding scheme of DCARPS. In this method, when a node receives a packet, it randomly chooses

one of many paths along which to forward it. In order to achieve this objective, we must slightly modify the path establishment process of DCARPS.

We propose two modified versions of the DCARPS protocol. The first one called *multipath DCARPS* is described in the next section and the second one called *randomized DCARPS* is described in section V.2. The former protocol has the advantage of allowing some control in terms of trading security vs. QoS. However, the second protocol is much simpler specially in terms of path establishment. In addition to the higher overhead, the path setup process of the probabilistic DCARPS requires a method for assigning multiple end-to-end paths to each node. In the next section, we propose an algorithm for this purpose which assigns to each node all of the possible end-to-end paths between that node and the sink. But, due to the large number of available paths as the network size grows, this algorithm is not scalable. In fact, our simulations could not be run for more than 20 nodes. A technique to limit the number of paths per sensor while keeping all the paths across the network consistent is needed. We feel that this is an interesting mathematical problem which can be addressed in future.

On the other hand, the randomized DCARPS protocol is very simple, scalable and efficient. Instead of using end-to-end paths, each forwarding node determines its next hop based on a local randomization of its available outgoing labels. In the next section, we propose the multipath DCARPS protocol. In section V.2, we describe and analyze randomized DCARPS using simulations.

V.1 MULTIPATH DCARPS

In a well connected network, there may be multiple routes connecting a sensor to the sink; the shortest path for that sensor is only one of these routes. We call a route starting at a leaf node a *path*. A leaf node is a node with no downstream neighbors on the shortest-paths tree. Leaf nodes are normally, but not necessarily, situated on the edge of the network. A path traverses many nodes and a node may belong to many paths even though it may only have one upstream link (i.e. one outgoing label) because a split anywhere in a route constitutes different paths. Therefore, an upstream link at a node may correspond to many paths. In the **multipath** DCARPS, the sink assigns a unique identifier to each path, which we refer to as a *path_id*.

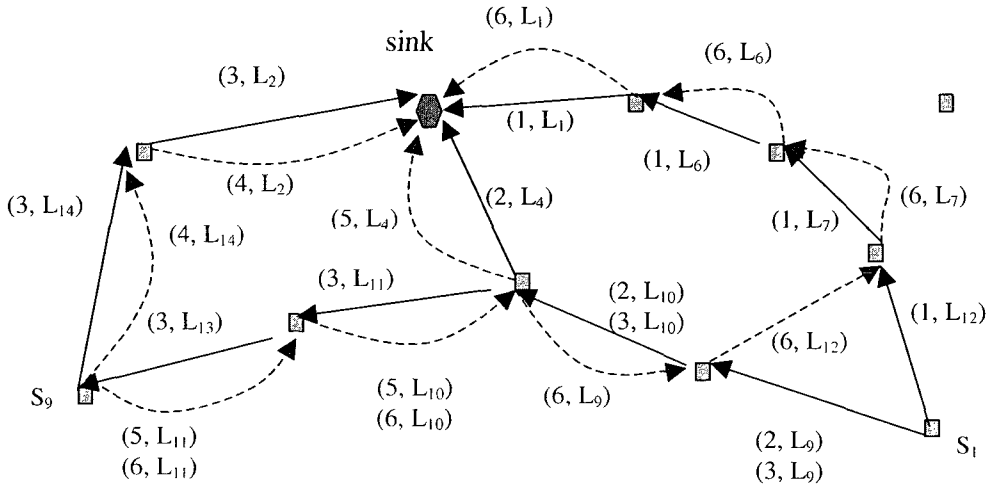


Figure V-1 Multipath path assignment for S_1 and S_9

Figure V-1 demonstrates how a link may belong to multiple paths and how a node may have multiple outgoing paths. In this figure, the solid lines and the dashed lines represent

paths originating at leaf nodes S_1 and S_9, respectively. The values in the bracket next to a link are the path_id and the upstream label associated with that link respectively, while the arrows show the direction of the uplink data packets for each path. A path_id uniquely identifies an end-to-end path while a label uniquely identifies a local receiver. A combination of a path_id and a label uniquely identifies a physical link.

In the original DCARPS path setup protocol, the sink assigns only one outgoing label to each sensor, regardless of how many incoming labels it may have. In essence, all the packets coming from different downstream neighbors would be forwarded to the same upstream neighbor (usually on the shortest path). In the probabilistic DCARPS, we assign outgoing labels for all of the neighbors of a sensor. When forwarding a packet, a sensor can randomly choose an outgoing label, preventing all packets from following the same path. However, if the label selection is totally random, packets may wander endlessly in the network, never arriving at the sink. To solve this problem in multipath DCARPS, we make sure that at each node, the packet is only forwarded on a route that has a smaller distance to the sink than when it arrived. This way, the packet is guaranteed to arrive at the sink in a bounded number of hops because the remaining distance for each packet is strictly decreasing. In randomized DCARPS, a different method is used. This method is presented in section V.2.

During the path establishment phase of multipath DCARPS, the sink informs each node of its incoming and outgoing labels as well as the paths to which it belongs. For each path_id, it lets the node know its distance from the sink on that path in terms of hop-counts. The sink also specifies the correspondence between the paths and the outgoing labels. For each sensor, multiple paths may have the same outgoing label (same parent).

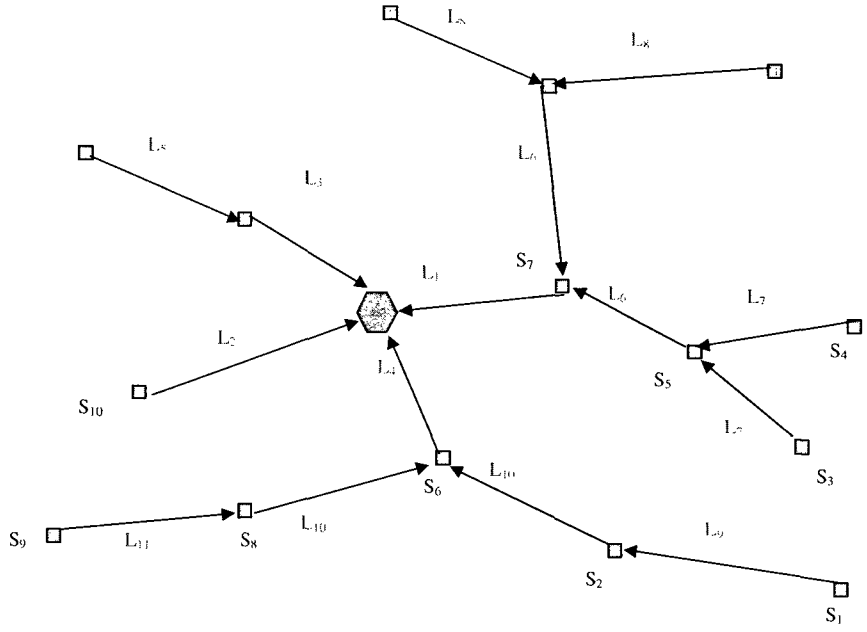


Figure V-2 Label assignments in a shortest-paths tree

Similar to the basic DCARPS, path_setup messages are created using layered cryptography and are distributed along the branches of the shortest-paths tree. However, in each layer of the path_setup onion, instead of one outgoing label, there will be one or more 3-tuples, each containing a path_id, the associated hop-count and the associated outgoing label, respectively. For example, assuming the paths of Figure V-1, the onion broadcasted on the main branch labeled L_4 is structured as follows. Please also refer to the label assignments in Figure V-2. Some nodes are not shown in Figure V-1.

$[DI_6, K_6(L_{10}, \{2,1,L_4\}, \{3,4,L_{11}\}, \{5,1,L_4\}, \{6,5,L_9\},$ *layer1*

$[DI_2, K_2(L_9, \{2,2,L_{10}\}, \{3,5,L_{10}\}, \{6,4,L_{12}\},$ *layer2*

$[DI_1, K_1(-, \{1,4,L_{12}\}, \{2,3,L_9\}, \{3,6,L_9\}))]],$ *layer3*

$[DI_8, K_8(L_{11}, \{3,3,L_{13}\}, \{5,2,L_{10}\}, \{6,6,L_{10}\},$ *layer2*

$[DI_9, K_9(-, \{3,2,L_{14}\}, \{4,2,L_{14}\}, \{5,3,L_{11}\}, \{6,7,L_{11}\}))]]]$ *layer3*

The second and the fourth lines describe the two sub-onions in layer#2, each one having another sub-onion of their own in layer#3. As can be seen, for large networks the packet size may become too big and require segmentation. The overhead of these messages is one of our motivations for proposing randomized DCARPS in section V.2.

During uplink data transmission, an incoming packet carries an incoming path_id as well as an incoming label, both unencrypted. The node chooses an outgoing path only among the paths that have a smaller distance than the incoming path. It makes this decision according to a certain probability density function $p(x)$. Then, it includes this path_id and the associated outgoing label in the packet in place of the old path_id and incoming label. Inclusion of a label is necessary because due to the broadcast nature of wireless networks, a data packet with a certain path_id may arrive at a node from an upstream or a downstream neighbor. If it is from an upstream neighbor it has to be ignored, otherwise it has to be repeated.

With this strategy, packets are distributed over different source-to-sink paths and a single adversary will have a difficult time tracking successive packets back to the sender. Indeed, at each sensor, an eavesdropper may have to wait for a long time for a future packet to arrive. However, since the remaining hop-count for a packet is always guaranteed to be strictly decreasing, the packet will arrive at the sink in a bounded number of hops. The upper limit on the number of hops that a certain packet may go through is the length of the path chosen by the source node at that node. It will also be hard for collaborating adversaries to locate the sender as long as they are not deployed too densely. Because, at different times, individual eavesdroppers may report that they have observed packets, but they are scattered across the network. Moreover, the packets

are assumed to be indistinguishable, thus two eavesdroppers cannot be sure that they have observed the same packet, which would mean they are both along the same route. Side-benefits of multipath routing include fault tolerance, load balancing (network longevity) and node capture tolerance.

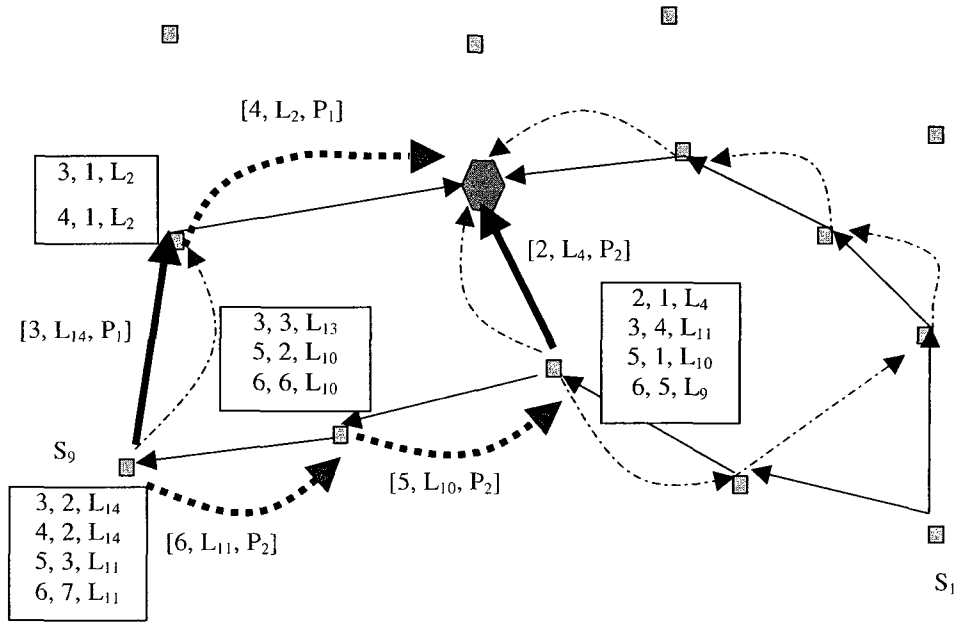
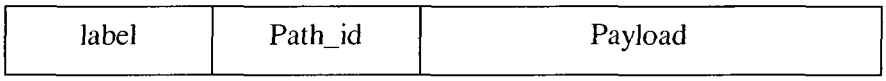


Figure V-3 Multipath packet forwarding

Figure V-3 depicts the transmissions of two packets, P_1 and P_2 , from S_9. The routing table of each node is shown next to it. On each row a path_id, its distance and its associated label are shown. Thick arrows (dashed or solid, blue or black) show the paths taken by the packets. As before, the solid lines show the paths starting at S_1 while the dashed lines show the paths starting at S_9.

In the previous chapter, we mentioned that the payloads of the uplink and downlink packets may have to be padded to make all packets the same length. If padding is done correctly, an adversary will be incapable of distinguishing between the encrypted payload in the downstream packets and "path_id + encrypted payload" in upstream packets. Figure V-4 shows the new format of the downstream and upstream packets. Because the DI label and the path_id do not share the same field in the packet formats, they can share the same pool of labels.

a. Downlink – control packet



b. Uplink – data packet

Figure V-4 Packet formats in multipath DCARPS

V.1.1 Security vs. Efficiency

A forwarding node may choose its outgoing path according to any probability density function (pdf) including a simple uniform distribution. However, p(x) may also be optimized for either more privacy or more efficiency (in terms of energy and latency). If we assign a higher probability of being selected to paths with larger distances (but still shorter than the incoming path), then the final path will converge to the shortest path

slowly and the packet will have a longer delay because it has more room to maneuver within the network. Note that at each hop, the number of available alternative paths from which to choose decreases and we merge to the shortest path between the current node and the sink. In other words, the average rate at which the size of the sample space for possible paths decreases along the path influences the achievable path diversity and is an indication of how much we are distancing ourselves from fixed-path routing.

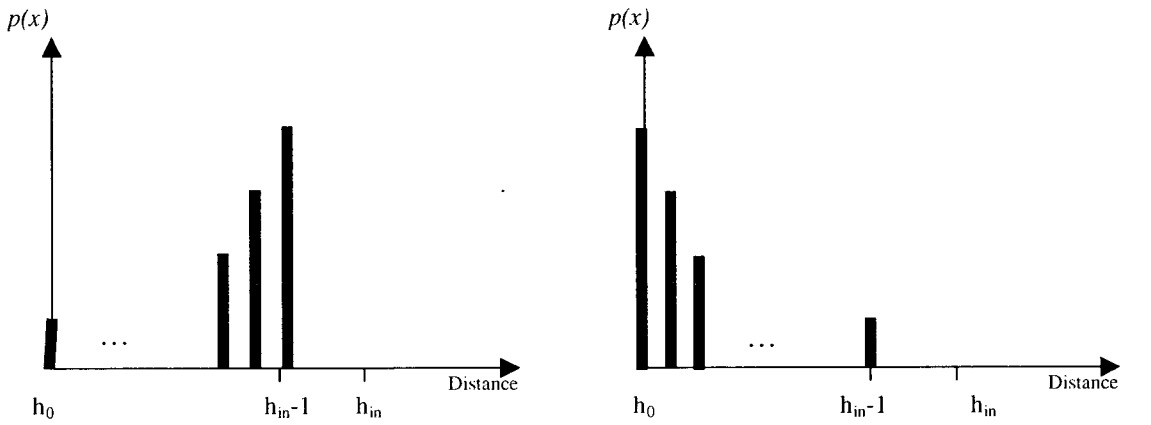
The variance of p(x) is another indication of how close we are to fixed-path routing. If the pdf is strongly concentrated in a small region of hop-counts, we will merge to fixed-path routing with a path length in that region. Figure V-5 shows two simple examples for this pdf. In this figure, h_{in} denotes the incoming hop-count carried in the packet and h_0 denotes the distance from the sink on the shortest path. The following equation must always be satisfied:

$$\sum_{x=h_0}^{h_{in}-1} p(x) = 1$$

Obviously, p(x)=0 for those values of x that do not exist i.e. when no paths with that distance are available to the sensor. At any node, this pdf varies for each incoming packet because it depends not only on the location of the sensor but also on the path_id of the packet. As an example, a sensor may use a finite geometric series to generate this pdf for an incoming packet.

Let's assume that $S = \{h_0, h_1, \ldots, h_k\}$ is the set of distances that are available to the sensor that are equal to or greater than the distance on the shortest path and smaller than h_{in}. The members of this set are arranged in an increasing order and each member

may represent one or more available paths. As was mentioned, h_0 is the distance on the shortest path while h_k is equal to h_{in} - 1. The size of S is |S|=k+1. The terms of a geometric series satisfying the afore-mentioned equation can be found from the following relationships.



a. Converges to the shortest path slowly.

b. Converges to the shortest path quickly.

Figure V-5 Examples for p(x)

For a slow-convergence pdf such as Figure V-5a:

$$\begin{cases} |S|=1, & p(h_0)=1 \\ |S|>1, & p(h_i)=\frac{1}{2^{k-i+1}},\ p(h_0)=p(h_1)=\frac{1}{2^k}, 0<i\le k \end{cases}$$

While for a fast-convergence pdf as in Figure V-5b:

$$\begin{cases} |S|=1, & p(h_0)=1 \\ |S|>1, & p(h_i)=\frac{1}{2^{i+1}},\ p(h_k)=p(h_{k-1})=\frac{1}{2^k}, 0\le i<k \end{cases}$$

If there are multiple paths with the same hop-count, one of them may be selected randomly.

There is a chance that two neighbors may belong to multiple paths going in the two opposite directions. Therefore, a packet may be passed from one node to the other on one path just to be returned on another path because it has a smaller distance to the sink. This event may be repeated several times, trapping the packet in a loop. But, this will not be an infinite loop because every time the hop-count for the chosen path must be less, eventually bringing the packet out of the loop. This event actually adds to the adversary's confusion as he must go back and forth between these two nodes.

V.1.2 Path Assignment

After the topology discovery phase and before path establishment, the sink must calculate and assign multiple paths to each node. However, in a large network, each sensor may have too many paths to the sink. Maintaining the information pertaining to all of these paths may require more memory than a sensor can afford. Moreover, communicating this data from the sink to the node consumes energy and bandwidth. To ease these burdens, we may put a cap on the number of paths per sensor. For example, the sink may choose a maximum number of paths per sensor that are as much as possible disjoint or are the shortest paths. In order to ensure connectivity, a path that is kept for a leaf sensor must be kept for all the nodes residing on that path. This requires the sink to ensure that the retained subsets of possible paths for all of the nodes are consistent with each other so

that each path is actually connected end-to-end. However, this presents a difficult mathematical and computational challenge.

Below, we provide an algorithm that assigns all the possible paths to every node. However, our simulations showed that this algorithm does not scale and is not viable for more than 20 nodes. It must be augmented with a method to keep the number of end-to-end paths per sensor limited. Similarly to the SPTree algorithm in DCARPS, this algorithm runs over the topology table, one tier at a time, in the order of increasing distance from the sink. However, in each tier, multiple iterations may be needed depending on the number of links between the nodes within that tier (intra-tier links). In each tier, the sink assigns all the neighbors of each node (including the ones on the same tier connected to it via intra-tier links) that do not belong to a higher tier (closer to the sink) as its children. Therefore, each node may also have multiple parents. This is unlike the SPTree algorithm.

The algorithm starts by assigning a different path-id to each one of the neighbors of the sink which constitute the first tier. Then, for each tier and in every iteration, the paths that have recently been determined to go through a node are extended to its children. Each path can be extended with the same path-id to only one child. Therefore, for other children of a node inheriting from that path, a new path-id is generated. We may impose a maximum length for each path by limiting the number of times it can be extended[14]. Therefore, a path may or may not end at a leaf node. A leaf node is a node that does not have any downlinks on the SPTree.

[14] This restriction is placed for keeping the number of paths small.

The intra-tier links are the key to having paths of different lengths per node. Without them, a node would have the same distance from the sink on all of its paths. We have intentionally decided to forgo the links to higher tiers in order to limit the complexity of the algorithm and make it converge faster. In the first iteration on each tier, every node extends to its children the paths that it has inherited from its immediate neighbors. However, because of intra-tier links, a node may inherit new paths from its neighbors on the same tier when its own turn in the iteration has already passed. Therefore, additional iterations are required to account for this kind of paths. Because the number of intra-tier links in each tier is limited, the process will eventually complete and the algorithm will move on to the next tier.

This algorithm consists of two stages. The previous paragraphs described the initial stage which calculates all the paths. However, the path assignments will not be complete without a second stage which we refer to as *recap*. As we mentioned before, when extending the paths of a node, the sink creates new path-ids if the node possesses more than one child. However, because the paths are end-to-end connections, the higher tier nodes on the extended path must also be assigned the new path-id. In order to be able to back track each path in the web of paths that it leaves behind, the sink maintains a list of previous hops for each path at every node. This is similar to route requests in a reactive routing protocol except all of this is done locally at the sink and no network communication exists at this point. This list also helps prevent loops due to intra-tier path extensions. An intra-tier extension is only allowed to neighbors of a node that do not exist in the list of previous hops of that path and that are not on the higher tier.

The recap stage starts after all the nodes have been processed in the initial stage and all the necessary paths have been created. The sink maintains a table of path-ids (*paths* table) that specifies the nodes residing on each path in the order of increasing distance from the sink. Therefore, in the recap stage, for each path-id, the sink makes sure that it is listed amongst the paths belonging to its constituent nodes in the topology table. If it does not exist, it will be added with the corresponding distance and the uplink label according to the SPTree.

In order to accommodate this algorithm, in addition to the paths table, a modification in the topology table is needed. Instead of the uplink label, this table will hold all the paths for each node in a column referred to as the *paths* column. Each path in this column consists of a 5-tuple as follows:

(path-id, path, distance, label, extend)

"path" is the list of previous hops on that path before this node. "d*istance"* shows how many hops this node is away from the sink on this path. "l*abel"* is the incoming label of the parent of this node on this path. "e*xtend"* is a Boolean flag that is set to TRUE when a new path is added to this node from a neighbor on the same tier indicating that this path has yet to be extended to the children of this node in the following iteration. In fact, a new path is always added to a node with this flag set to TRUE. When the path is extended, this flag is set to FALSE. Once there are no extend flags that have a TRUE value in a tier, the algorithm moves on to the next tier. Algorithm V-1 below describes this process.

Algorithm V-1 Path_id assignment in multipath DCARPS

Procedure ***multipath_path_calc ()***

Input*: topology table*

Output*: end-to-end paths*

Begin

For *all Tiers*

While *there exists an extend flag =TRUE*

For *all nodes*

Extend all extendable paths to one eligible neighbor

// eligible means not in the "path" or a higher tier

// keep the same path_ids

Extend all extendable paths to other eligible neighbors

// generate new path_ids

// Do not extend a path beyond the maximum permitted length.

// Recap

For *all paths in the paths table*

For *all nodes on the path*

Add path to the node's paths in the topology table.

End

In order to avoid the complexity of assigning end-to-end paths to every node, in the next section we propose a different method for spreading a data flow over multiple paths while ensuring that packets do not wonder in the network aimlessly.

V.2 RANDOMIZED DCARPS

In this version of the probabilistic DCARPS, we allow a forwarding node to choose any of its outgoing labels randomly for the next hop. However, in order to ensure that a packet reaches the sink in a reasonable number of hops we consider a limit on the number

of hops that it can be forwarded randomly in this manner. Indeed, we include a Time-To_Submit (TTS) field in each data packet which is decremented every time it is forwarded randomly. Once this value reaches zero, the packet is forwarded on the shortest path towards the sink. In order to prevent an adversary from guessing the location of the source node from this information, the source initializes this field of every data packet to a different random value.

The path setup messages of this protocol are much simpler than the multipath DCARPS. For the purpose of illustration, the path setup message sent on the main branch L_4 in figure V-6 is shown below. One thing to note is that in randomized DCARPS, every node including the leaf nodes are assigned an incoming label and may receive data packets from its neighbors to forward.

$[DI_6, K_6(L_{10}, \{L_4, L_9, L_{11}\},$ *layer1*

$[DI_2, K_2(L_9, \{L_{10}, L_{12}, L_{15}\},$ *layer2*

$[DI_1, K_1(L_{15}, \{L_{12}, L_9\}))]],$ *layer3*

$[DI_8, K_8(L_{11}, \{L_{13}, L_{10}\},$ *layer2*

$[DI_9, K_9(L_{13}, \{L_{14}, L_{11}\}))]]]$ *layer3*

The first label in each layer is the incoming label of the recipient node while the labels in the curly brackets represent its outgoing labels. The data packets have a similar format as in the multipath DCARPS shown in figure V-4 except that in the uplink direction they carry a TTS field instead of the path_id. The routing information maintained by each node is simply a set of available outgoing labels.

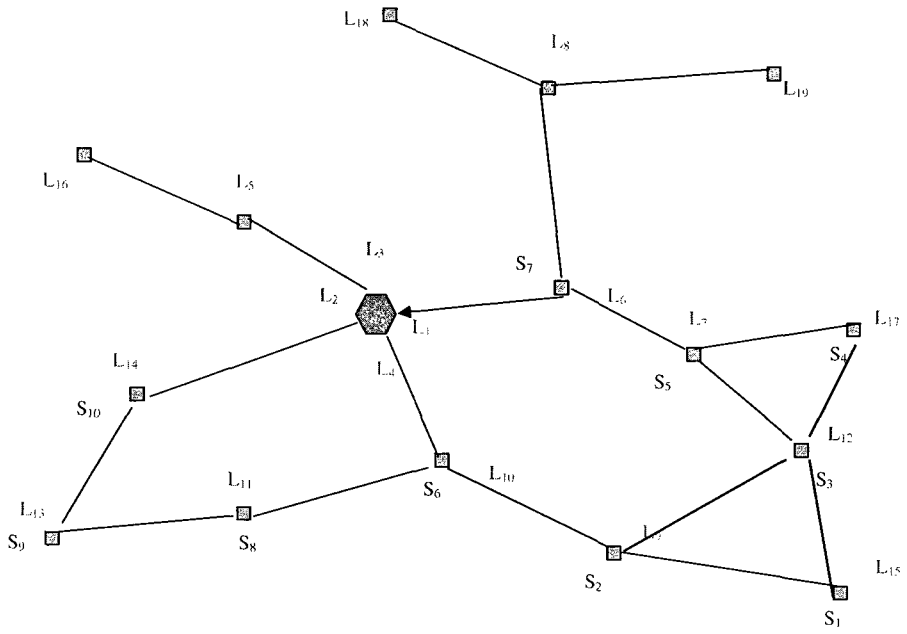


Figure V-6 Label assignments in randomized DCARPS

In our implementation of this protocol, the selection of the next hop at a forwarding node is done purely randomly according to a uniform probability distribution. But, we suggest that the security performance of the protocol may be improved by introducing intelligence in this task. For example, it is desirable to move packets away from the source as quickly as possible and prevent them from lingering within a limited area. However, a forwarding node does not have any information regarding the relative locations of its neighbors. It only knows their incoming labels which are actually its outgoing labels.

One way to achieve this goal is for the sink to indicate a relative direction for each outgoing label of a node using the path setup messages. For example, the sink can arbitrarily consider any direction to be north and then determine the relative direction of

the link between a node and each one of its neighbors accordingly. We can consider 4 geographic directions namely north (N), south (S), east (E) and west (W) or 8 directions by adding NW, NE, SW and SE. When generating a data packet, the source node chooses one of these directions randomly and indicates that in a special field in the packet. Each forwarding node must try to select one of its outgoing labels that is closest to the original direction of the packet while the TTS field is not zero.

V.2.1 Simulations

We implemented randomized DCARPS in OPNET simulator and compared its security performance against regular DCARPS and flooding in the presence of a single source node (the extreme case of a scarce-traffic network[15]) and a location-limited eavesdropper. We considered the hearing range of the eavesdropper to be the same as the transmission range of sensor nodes. Obviously, the larger the hearing range of the adversary the stronger it is. Sensors were distributed in an area of 100 x 100 according to a uniform probability distribution. We experimented with network sizes of 50, 60, 70, 80 and 100 nodes and for each network size we conducted simulations for 20 different random network topologies. For a certain network size and a certain network topology, we repeated simulations for various scenarios by changing the source node and the initial location of the eavesdropper. Specifically, we tried 10 different source nodes for each network topology of size 50 or 60 and 20 different source nodes for each network topology of size 70, 80 or 100 nodes. Furthermore, for each of these scenarios, we tried 5

[15] A bigger number of source nodes makes the task of the adversary more difficult.

different initial locations for the eavesdropper. Therefore, a total of 8000 simulation runs were performed for each protocol.

In order to compare different protocols, we used *safety period* as the performance metric defined as the time (in terms of number of packets generated by the source node) it takes the adversary to locate the source node. In all of our simulations, the source node followed the same traffic generation pattern, issuing one packet per second on average. The source node is considered detected as soon as the adversary receives a packet directly from it. The TTS field in every packet is initialized to a random value. We used a uniform random distribution in the range of 0 to 15 for network sizes of 50 and 60 or 25 for network sizes of 70, 80 and 100. A TTS value of 0 represents the regular DCARPS protocol.

Eavesdropper Strategy

The local eavesdropper launches a packet tracing attack by moving to the immediate sender of each data packet that it receives. However, if it does not receive any packets for a certain amount of time, it moves to a new randomly selected location and again awaits the packets. The wait period in our simulations was chosen to be almost equal to the time needed for generation of 5 packets and the distance that it traveled was equal to the nodes' transmission range. The eavesdropper moved along one of 4 perpendicular directions but it never moved back to its immediate previous location and it never moved out of the network area.

Results

Figure V-7 compares the safety period for our randomized DCARPS protocol vs. flooding as well as regular DCARPS protocols. One thing that has to be noted is that for each one of the three protocols, we have considered the aggregate results obtained from all of the simulations. Due to the dependence of the eavesdropper motion pattern on the routing protocol (it moves to the location of a node from which it receives a packet and also changes location if it does not receive a packet for a while), no two specific scenarios can be compared because the overall path of the attacker most likely does not repeat. Therefore, the safety period for each protocol is the average of the safety periods obtained from all of its simulation runs.

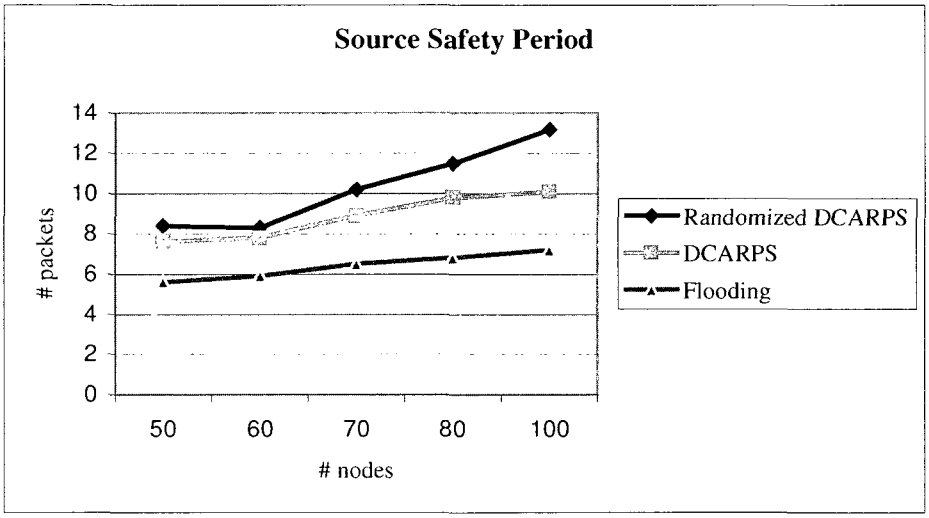


Figure V-7 Source safety period of randomized DCARPS vs. flooding and DCARPS

As can be seen, flooding has the shortest safety period. This is expected because every data packet is received by the eavesdropper and most likely on its shortest path to the

sender first. Also, the advantage of randomized DCARPS over DCARPS increases with an increase in the number of nodes. This is also expected as this protocol relies on the route diversity which is more pronounced for larger networks. In other words, as the network size grows the advantage of this protocol over the regular DCARPS becomes more obvious.

Similar results are found for the destination safety period because the eavesdropper strategy is very similar as explained in previous chapters. It waits for a packet arrival from a local node and then it waits for the transmission of the same packet from another neighboring node. Then, it moves to the location of the second transmitting node. However, in this case, flooding shows perfect destination privacy because the sink also repeats every packet thus it is not distinguishable from other nodes.

V.3 SUMMARY

In this chapter, we proposed two probabilistic routing protocols based on DCARPS that provide location privacy and anonymity for wireless sensor networks which carry a small number of data flows at any given time. These protocols protect against a local eavesdropper since protecting such a network against a global eavesdropper is impossible.

As opposed to the basic DCARPS that uses fixed-path routing, the probabilistic DCARPS protocols route every packet of a data flow randomly over diverse end-to-end paths thanks to node-level probabilistic packet forwarding schemes. This strategy prevents a location-limited attacker from receiving every packet from the same stream one after another and thus advancing one hop per packet towards its target. This way, the

safety period is prolonged i.e. the attacker will spend a longer time trying to track down its target.

Two different versions of the probabilistic DCARPS were proposed. The multipath DCARPS protocol makes use of randomly selected end-to-end paths by the source node and each forwarding node. On the other hand, the simpler randomized DCARPS protocol relies on local randomization of outgoing labels by each forwarding node. Our simulations showed that this protocol outperforms basic DCARPS in terms of source and destination safety period while it also outperforms flooding in regards with source safety period.

The randomized nature of these routing protocols makes them resilient to attacks such as "traffic pattern analysis" which detect the most used routes in the network. Moreover, they promote load balancing and fault tolerance in the network by spreading the traffic over different paths.

CHAPTER VI

EFFICIENT NETWORK TOPOLOGY DISCOVERY

In this chapter, we propose a low-overhead topology discovery protocol for wireless networks. This protocol is based on the link state approach and is called 2hop-clustered NTDP. It is intended for use during the topology discovery phase before the V-routing or DCARPS protocols can be used. The main motivation for this chapter is reducing the overhead of topology discovery in proactive routing protocols so that they can be used more efficiently in resource constrained wireless ad hoc and sensor networks

---

As we discussed in previous chapters, we have used proactive routing approaches as a cornerstone for all of our protocols for both V-routing and DCARPS families. The reasons for this adoption were mentioned to be mainly the slow-start problem of reactive routing protocols and also suitability of the proactive approach for preserving location privacy, among other reasons. However, in those discussions we also mentioned that the overhead of periodic topology updates for network topology discovery in proactive routing protocols has always been the main impediment in the way of using this class of protocols in wireless ad hoc networks specially sensor networks. At the same time, we pointed out that a lot of progress has been made recently in improving the performance of these protocols. There are 4 major directions for dealing with the overhead associated with the periodical topology updates.

1- Limited dissemination techniques such as MPR (MultiPoint Relay) are used to reduce the flooding overhead associated with topology update messages[16].

2- Some protocols such as those proposed in [131] attempt to reduce this overhead by tuning the value of refresh intervals dynamically and automatically so that topology updates are triggered based on traffic conditions and node mobility.

3- There are also approaches like ZRP [27] that combine the proactive and reactive strategies in order to limit the scope of topology update messages i.e. the area in which they are flooded.

4- The last method is reducing the size of topology update messages. These protocols can be divided into two categories themselves. The first category includes partial topology update techniques. For example, in OLSR, each node broadcasts only the list of its neighbors that have selected it as an MPR. In TBRPF, a node calculates a source tree to every reachable destination but then broadcasts only part of it. The second category, as far as we know, has only one member and that is the protocol that we introduce in this chapter. This protocol reduces the redundancies in the topology update messages.

Our protocol called *2hop-clustered Network Topology Discovery Protocol (NTDP)* is based on the link state approach for network topology discovery and uses node clustering and exploits bi-directional links in order to reduce the overhead of topology update messages. In this protocol, only clusterheads (cluster leaders) generate topology update packets as opposed to ordinary link state protocols where all nodes do so. In addition,

[16] Gateway flooding is another limited flooding technique but MPR has been shown to have a better performance.

clusterheads remove the redundancy associated with the double inclusion of bidirectional links in topology updates as we will explain later. Our simulations show over 50% reduction in the control overhead of topology discovery compared to a regular link state routing protocol.

The other category of proactive routing protocols namely distance vector (e.g. DSDV) has not enjoyed the attention that the link state approach has received in the context of ad hoc routing protocols. Incidentally, our redundancy reduction technique is not applicable to this class of proactive routing protocols as it does not perform true network topology discovery. Instead, each node only computes and broadcasts its distance from every other node in the network as opposed to its neighborhood information as is done in link state.

Aside from routing protocols, other applications of link state topology discovery protocols include central network management and topology control [124-126] in the context of multihop wireless sensor networks. In a WSN, these protocols enable the sink to monitor, control and co-ordinate node activities (e.g. active/sleep cycles) in the network in a manner that maximizes the overall network lifetime.

VI.1 PROBLEM STATEMENT

An essential component of any link-state routing protocol (e.g. OSPF [86], OLSR [28]) is a Network Topology Discovery Protocol (NTDP). Using an NTDP, a node can gain knowledge about the active nodes and the existing links in the network. In these protocols, normally each node in the network broadcasts periodic messages in the network containing its identifier (ID) and information about its neighborhood,

particularly the identifiers of its neighbors. This process is depicted in Figure VI-1. In the link-state routing terminology, these messages are referred to as *topology updates* or *Link State Advertisements (LSA)*. By combining LSAs from all other nodes, any node in the network can maintain an up-to-date view of the global network topology. However, these messages also introduce considerable communication overhead and energy consumption.

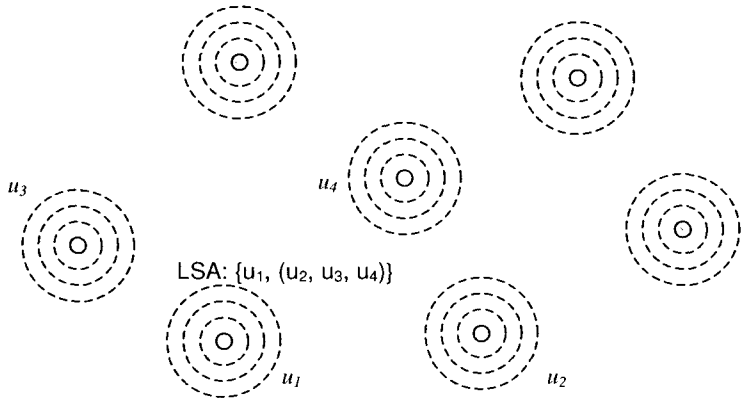


Figure VI-1 Routing updates in link state topology discovery

Energy conservation is a very important issue in the design of protocols for wireless networks and transceivers are the main consumer of battery in wireless devices. For example, the total energy consumed for the transmission and reception of a packet using a MICA mote is almost 2.5 times that of computations' [122]. Moreover, due to the spectrum limitations in wireless networks and small bitrates of wireless devices, bandwidth is at a high premium too. Hence, there has been a lot of effort in reducing the costs associated with broadcasting routing advertisements in wireless ad hoc networks. In particular, the MultiPoint Relay (MPR) technique [132], used in OLSR, has been very

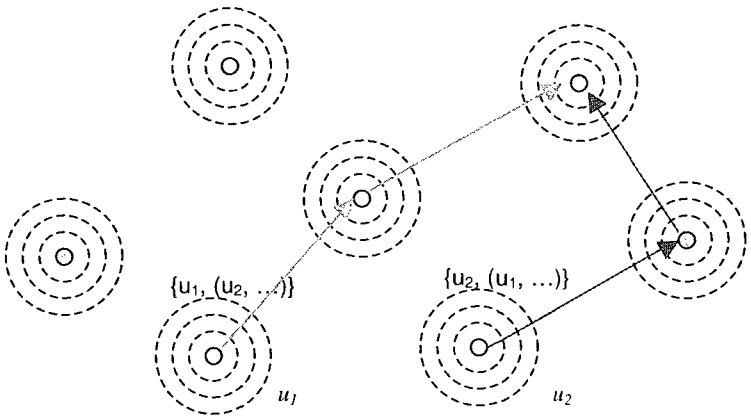


Figure VI-2 Redundancy in LSAs originated by neighboring nodes

successful in reducing the flooding overhead by limiting the number of nodes that are allowed to repeat a flooded packet while ensuring that every node in the network will receive a copy of that packet. Several protocols have tried to limit the scope of dissemination of these messages in space and time. One such algorithm and a review of some other algorithms in this category can be found in [133]. However, much redundancy still remains within the original topology update information itself. This is due to the fact that each bi-directional link is reported twice, once by each of the two ends of the link. A link between two nodes is said to be bi-directional if each node can hear the other node's transmissions. Essentially, each end of a bi-directional link identifies the other end as its neighbor within its topology update messages. For example, in Figure VI-2 the two nodes u_1 and u_2 each declare the other one as neighbor in their LSAs. A remote node needs to receive this information only once because adjacency in a bi-directional link is a mutual relationship.

VI.2 2hop-Clustered NTDP

In order to reduce the overhead of regular link state topology discovery protocols, in our protocol we use an information compression mechanism that eliminates duplications across individual topology update messages. If all the links in the network are bi-directional (a common assumption), this method can theoretically improve the performance of an NTDP by at least 50%. The amount of savings is actually more because we not only eliminate the duplication of bi-directional links but also reduce the number of originated topology update packets using a clustered architecture explained below.

We use a clustered architecture whereby nodes are dynamically organized into clusters and only one node in each cluster, called the *ClusterHead (CH)*, generates a topology update packet containing the information regarding the links that involve itself and all the *ordinary* (non-CH) nodes in that cluster. As will be explained, neighboring CHs co-ordinate their efforts in a way that inter-cluster links are reported as well, allowing other nodes in the network to have complete knowledge of the network topology in that locality. Please note that this clustered architecture is only used during topology discovery not the actual data transmission phase.

In our 2hop-clustered NTDP, only one node in each cluster generates topology updates instead of every node and since each such message must carry the ID of the sender, a lot of savings is achieved that way. The price that is paid to obtain these savings is a limited amount of processing at the sender and the destination nodes. These computations consist of simple table lookup operations which can be efficiently implemented. In wireless sensor networks, there is usually only one destination for the

topology updates, namely the sink, which normally is not restricted in terms of processing power and energy. In case of the sender, as we said before, computation spends much less energy than transmission. Typically, transmitting 1 bit consumes as much energy as executing 1000 instructions. Furthermore, the overall positive impact of our method on energy conservation is more profound as it is not confined to the sender since each transmitted bit must also be received (which is also expensive) and retransmitted by intermediate nodes.

VI.2.1 Protocol Description

Consider a network of randomly distributed nodes. Although our protocol is easily applicable to wired networks, here we will assume a wireless network without loss of generality. Furthermore, we do not assume tight time synchronization among the nodes. In the following, we describe various parts of our protocol.

Clustering

We model our network as an undirected graph G (V, E) where V denotes the set of nodes (vertices) and E denotes the set of links (edges) in the network. At this time, we assume only bi-directional links but the protocol can be slightly modified (with negligible extra overhead) to support uni-directional links. Upon activation, similar to other link-state routing protocols, each node broadcasts an *initial HELLO* message locally. These messages contain only the ID of the sender and allow the node to be identified as a neighbor to the nodes that can hear it. After a while, when no more initial HELLO

packets arrive, a node enters the clustering state where it decides whether it becomes a clusterhead.

Clustering is a commonly used technique in wireless ad hoc networks that allows network nodes to dynamically and automatically organize themselves into groups usually for the purpose of achieving better performance. The applications of clustering range widely from routing protocols to security protocols and from code allocation in CDMA networks to in-network processing for sensor networks. An overwhelming body of research exists on clustering algorithms and schemes. However, most of these schemes have many things in common. For example, they all divide network nodes into groups called clusters where each cluster is represented by one of the member nodes called the cluster leader or the clusterhead. Clusterheads normally perform special extra tasks on behalf of the ordinary nodes in their clusters. The clusterhead is selected by the cluster members according to certain criteria.

Even though any desired clustering algorithm may be used in our protocol, we use the simple lowest-ID criterion [134], which selects a node as a CH if it has the smallest ID among its undecided neighbors and is not neighbor to another CH. A node is considered undecided if it has not yet determined if it becomes a CH or an ordinary node. The only restriction that we impose is that an ordinary node can be a member of one and only one cluster. In other words, we have strictly non-overlapping clusters. This clustering algorithm is in fact a special case of the MWIS (Maximal Weighted Independent Set) clustering algorithm that has been formally described and analyzed in [135], where it is proven that at the end each ordinary node will be a member of exactly one cluster. MWIS makes use of two kinds of locally broadcasted messages: CH(v) that

is sent by a node 'v' to announce that it has become a CH and JOIN(u, v) that is sent by an ordinary node 'u' to announce that it is a member of clusterhead 'v'.

In order to facilitate our clustering algorithm as well as the MPR algorithm later on, we introduce a new locally broadcasted *follow-up HELLO* message that incorporates the role of the CH(v) message of MWIS. As soon as its role is determined, a node v sends a follow-up HELLO message (v, Γ(v), ch), where Γ(v) is the set of neighbors of v and ch is a Boolean flag indicating whether v is a CH.

MPR Sets

In conjunction with our own protocol, we used the MPR technique in order to also limit the flooding overhead and achieve a better performance. We implemented the MPR set selection heuristic algorithm proposed in [132]. Using our follow-up hello messages, each node obtains complete knowledge of its 2hop neighborhood and is therefore capable of determining its multipoint relay set amongst its immediate neighbors. When all the nodes in the network choose their MPRs, any flooded message (in this case, topology updates) can be received by all the nodes with minimal overhead. However, in our protocol, only a subset of nodes needs to select MPR sets because only clusterheads generate packets that must be flooded in the network. Therefore, at first only CHs select their MPR sets. Then, any node that has been selected as an MPR node by a CH or another MPR will calculate an MPR set of its own. We refer to the final set of MPR nodes as the *selected MPR superset.* A node is counted in this set as many times as the number of its MPR selectors. A node y is called an MPR *selector* of node x if it has chosen x as one of its MPRs. This modification (that only some nodes need to select

MPRs) results in considerable savings in terms of messaging because every MPR selector must inform its MPRs of its decision to include them in its MPR set.

2hop Clusters

In our 2hop-clustered NTDP, clusterheads are responsible for generating topology updates and ordinary nodes (only if they are selected as MPR for other nodes) only repeat these messages. In order to reduce the size of these packets, clusterheads perform redundancy reduction operations trying to prevent a bidirectional link from being reported twice, as it would be in the case of regular link-state routing protocols. As mentioned before, if there are uni-directional links, this protocol will have to change slightly but the clusterheads are in the perfect position to identify such links because they have 2hop neighborhood information. In that case, the two kinds of links must be distinguished in the topology updates. A CH is responsible for reporting the links involving its members. In case of a conflict, the CH with the lowest ID wins, as we will explain here.

In a clustered architecture, two main kinds of links can be distinguished:

Intra-cluster links: Both connected nodes belong to the same cluster e.g. $<v_1, u_{11}>$ and $<u_{13}, u_{14}>$ in Figure VI-3.

Inter-cluster links: Each one of the two connected nodes belongs to a different cluster. If one of the two nodes is a CH (based on MWIS, no two CHs can be neighbors), we call that link a gateway link (because the other node usually plays

the role of a gateway between the two clusters during data transmission) e.g. $<u_{11}, v_2>$ in Figure VI-3, otherwise we call it a non-gateway link e.g. $<u_{14}, u_{21}>$.

In case of an intra-cluster link, there would be no conflict because only one CH is responsible for both nodes and will report the link only once. In case of a gateway inter-cluster link, both CHs receive the JOIN message sent by their common ordinary node and know which one is responsible for it. The CH with the higher ID does not report its link with that node or any other link involving that node.

In case of a non-gateway link, two situations may arise.

a. If the ordinary node belonging to the CH with the smaller ID also has a direct link with the CH in the opposite cluster, then that CH can realize that it is not responsible for that node because it receives its JOIN message. Therefore, it will refrain from reporting the non-gateway link while the other CH will include it in its

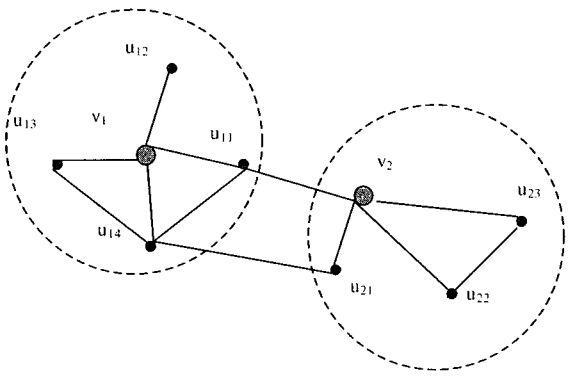


Figure VI-3 Various kinds of link in a clustered architecture

topology update. This way, the conflict is resolved and the non-gateway link is reported only once. Please note that the ordinary node belonging to the CH with the higher ID cannot have a direct link to the CH with the smaller ID because in that case it would become a member of that CH as an ordinary node.

b. If situation (a) does not occur, none of the two CHs receive the JOIN message sent by the ordinary node in the opposite cluster. Thus, they cannot coordinate their actions and both must report the non-gateway link. This problem could be remedied if we allow the JOIN messages to be repeated once so that the CHs can receive JOIN messages from their 2hop neighbors but the resulting overhead might negate the savings gained by the prevention of the double inclusion of that link. In our implementation, we have chosen to allow this kind of duplication.

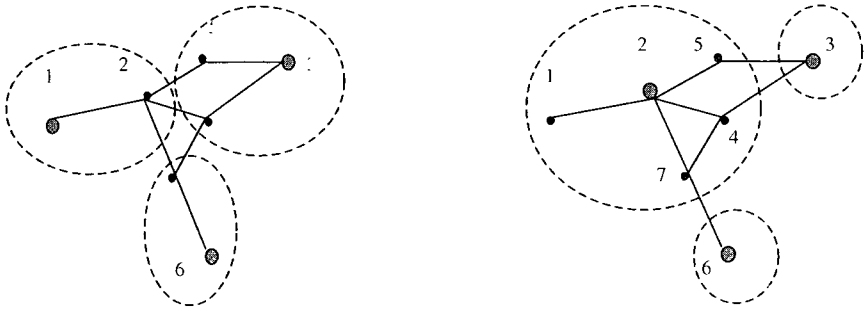
Proposition 1: Each link is reported at least once.

Proof: The proof follows directly from the previous paragraphs. If the link is an intra-cluster link, it connects an ordinary node either to its CH or to another ordinary node within that cluster. In both cases, the link belongs to only one CH and is reported only once. If the link is a gateway inter-cluster link, the CH with the smaller ID takes responsibility and it will be reported only once. If it is a non-gateway link and situation (a) mentioned above is true, it is reported only once by the CH with the smaller ID. Otherwise, situation (b) is true and the link will be reported twice.

Conclusion: The overhead of topology updates would be minimized if the number of non-gateway links is minimized.

The number of inter-cluster links depends on the employed clustering algorithm. It has been shown that optimal clustering is an NP-hard problem [136]. Therefore, finding a clustering algorithm that satisfies an extra constraint such as minimizing this number is also NP-hard. However, different heuristics may be compared. For example, it seems that a highest-degree clustering criterion (degree means the number of neighbors of a node) may produce more efficient cluster formations than the lowest-ID clustering because it maximizes the number of intra-cluster links since any link involving a CH would be an intra-cluster link. For instance, in the simple network of Figure VI-4, the lowest-ID clustering generates 4 non-gateway links (2-4, 2-7, 2-5, 4-7) while the highest-degree clustering generates none.

However, our experiments did not show a discernable improvement and we believe the reason is the influence of other factors in particular MPR. When limited flooding techniques such as MPR are used, especially if our afore-mentioned modification is also applied where only some nodes select MPRs for themselves, then the



lowest-ID highest-degree

Figure VI-4 Cluster formations

role of cluster formations becomes important. In the absence of these techniques, the only difference between two different clustering algorithms is the number of non-gateway inter-cluster links because all other links are reported exactly once by our topology discovery protocol and each topology update message is repeated by all the nodes. However, when MPR is used, topology update messages of different sizes (due to random cluster formations) are repeated by different numbers of nodes before they reach the destination. This phenomenon may help or negate the savings gained by the elimination of non-gateway links.

Moreover, the overall performance evaluation must take into account the possible increases in the messaging overhead required by the clustering algorithm. For example, in our implementation, the highest-degree clustering could be easily achieved by making the clusterhead decisions after the transmissions of all of the follow-up HELLO messages (currently, the nodes with the smallest ID in their neighborhood can decide that they become a CH before sending their follow-up hello messages) but then each CH must also submit a separate message announcing its role after it has sent its follow-up hello, which would add one ID worth of overhead to the total control traffic.

Proposition 2: If JOIN messages are repeated at least once, all possible clustering algorithms that comply with MWIS (i.e. produce an independent set and each ordinary node belongs to one and only one CH) generate the same amount of original (excluding the repeated packet) topology update overhead.

Proof: In this situation, regardless of the choice of clustering algorithm, a CH can always identify its 1hop and 2hop constituent links and the non-gateway inter-cluster links do not need to be reported twice. Therefore, every existing link will be reported exactly once.

Conclusion: The problem of finding the best set of clusterheads in this case reduces to maximizing the cardinality of this set and minimizing the cardinality of the selected MPR superset at the same time.

The former contributes to reduce the overall overhead by reducing the number of JOIN messages since the reduction in the number of ordinary nodes (and hence the number of JOIN messages) is equal to the increase in the number of CHs. The latter ensures that the amount of repeated traffic is minimized because only MPRs are allowed to forward packets. In this case, an upper bound on the sum of the control traffics for clustering and topology updates can be calculated as follows:

Denote the cardinality of the selected MPR superset with M, the number of nodes in the network with N, the average number of nodes in a 1hop neighborhood with n, the number of active links with l, the number of clusterheads with c, the number of links in the 2hop cluster i with l_i, $\forall i \in [1, c]$ and the number of MPRs for each MPR selector with m_j so that $\sum_j m_j = M$. The topology update generated by c_i is repeated first by its MPRs and ultimately by all of the other MPRs in the network. Therefore, it produces a total overhead of at most $\sum_j (l_i + 1)m_j = (l_i + 1)M$ in terms of IDs (one CH and l_i nodes in the i^{th} cluster). Please note that this is an upper bound because it considers that a packet is repeated by each MPR as many times as it has been counted in the selected MPR superset

while in actuality our implementation uses limited flooding which means each node repeats a certain packet at most once. Because each link is reported only once, the total overhead of topology updates is at most:

$$\sum_{i=1}^{c} l_i M = lM$$

The overhead of JOIN messages is equal to $2n(N-c)$ because each such message contains two IDs (the sender and its CH) and is broadcasted by an ordinary node and all of its neighbors. Therefore:

$$Total\ OH < lM + 2n(N - c)$$

Total overhead includes overheads associated with both clustering and routing updates. However, another factor has to be considered here and that is the extra messaging overhead required in order to achieve this optimization which is also implementation dependent.

Topology Update Messages

The constituent links of clusterheads v_1 and v_2 from Figure VI-3 are shown in Figure VI-5, assuming $v_1 < v_2$. The links belonging only to v_1 are shown thinner and in a lighter color compared to those belonging to v_2. The non-gateway inter-cluster link $<u_{14}, u_{21}>$ shown with the thickest line belongs to both clusterheads and is reported twice. As can be seen, the area of responsibility of a CH can be 2 hops wide hence the name "2hop clusters " (clusters of links not nodes).

In the 2hop-clustered NTDP, the link state advertisements have the following format:

$$\text{LSA}\ (v, u_1, \Gamma'(u_1), \ldots, u_n, \Gamma'(u_n))$$

where v is the sender of the packet, u_i is a neighbor of v, $\forall\ i \in [1, n]$ where n is the cardinality of $\Gamma(v)$ and $\Gamma'(u_i)$ is the reduced neighborhood of u_i (which are 2hop neighbors of v), namely all of its neighbors whose link to u_i is not included any where else in the packet.

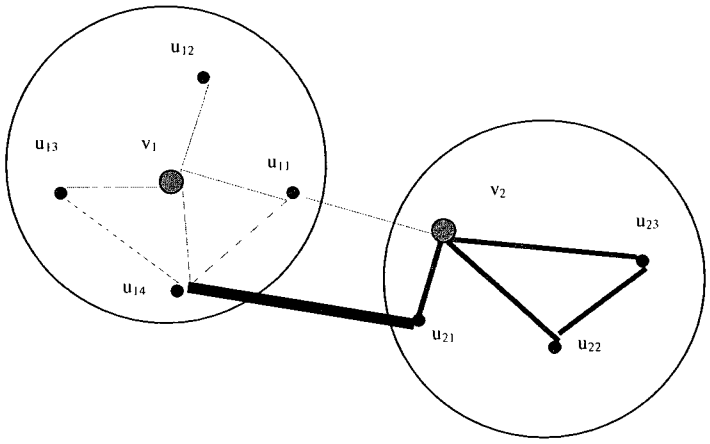


Figure VI-5 2hop clusters

In order to generate its LSA, a clusterhead v executes Algorithm VI-1. For example, v_1 and v_2 in Figure VI-5 produce the following LSA packets:

v_1: $(v_1, u_{11}, \{u_{14}, v_2\}, u_{12}, \{\}, u_{13}, \{u_{14}\}, u_{14}, \{u_{21}\})$

v_2: $(v_2, u_{21}, \{u_{14}\}, u_{22}, \{u_{23}\}, u_{23}, \{\})$

for a total overhead of 15 IDs while in a regular routing protocol each node would broadcast all of its neighborhood resulting in an overhead of equal to 33 IDs based on Table VI-1. As can be seen, the amount of savings is more than 50%.

Algorithm VI-1 LSA generation by a clusterhead

Procedure *generate_LSA;*

(Input: $\Gamma(v)$, LSA=Φ , $\Gamma'(u)$= Φ **For all** u **in** $\Gamma(v)$)

(Output: LSA(v, {u, $\Gamma'(u)$}) **For all** u **in** $\Gamma(v)$)

Begin

For all u **in** $\Gamma(v)$

Add u **to** LSA

For all w **in** $\Gamma(u)$

If u $\notin$ $\Gamma'(w)$ **Then**

Add w **to** $\Gamma'(u)$

End;

Table VI-1 Number of neighbors of nodes in Figure VI-5

node	v_1	v_2	u_{11}	u_{12}	u_{13}	u_{14}	u_{21}	u_{22}	u_{23}
#neigh	4	4	3	1	2	4	2	2	2

Updating Topology Information

At the destination, a node is able to successfully construct a complete image of the network topology using the reduced topology updates that it receives. The destination maintains a topology table that identifies the neighbors of any arbitrary node in the table.

Each row of this table includes two columns, one for the ID of a node and one for the IDs of its neighbors. When the destination node processes a topology update, it fills in not only the row corresponding to the sender but also the rows corresponding to its 1hop and 2hop neighbors. In other words, it expands the information in the packet based on the fact that all the links are bidirectional. In other words, if $x \in \Gamma(y)$ then $y \in \Gamma(x)$ .Algorithm VI-2 outlines this process. Please note that the destination may receive the same copy of an LSA multiple times but it only processes the first copy and discards the rest.

Algorithm VI-2 LSA expansion upon receipt of an LSA

Procedure *expand_LSA;*
(Input: LSA(v, { u , Γ'(u)}), top_tbl{. , Γ(.)})
(Output: top_tbl{. , Γ(.)})
Begin
    If v $\notin$ top_tbl **Then**
        Add v **to** top_tbl
    For all u **in** LSA
        Do *update (v,u)*
        For all w **in** Γ'(u) **of** LSA
            Do *update (u,w)*
End;

Where procedure *update ()* is shown in Algorithm VI-3.

Algorithm VI-3 Topology table update

Procedure *update (v,u);*

(Input: top_tbl{. , Γ(.)})

(Output: top_tbl{. , Γ(.)})

Begin

If u $\notin$ top_tbl (Γ(v))

Add u **to** top_tbl(Γ(v))

If u $\notin$ top_tbl

Add u **to** top_tbl

If v $\notin$ top_tbl(Γ(u))

Add v **to** top_tbl(Γ(u))

End;

For instance, when a node receives the updates from v_1 and v_2 of Figure VI-5, it realizes that u_{14} is a neighbor of v_2 even though it is not listed as such in the topology update sent by v_2. Part of the topology table at every destination node looks like Table VI-2 after these two packets are received. In wireless sensor networks, usually, the only node interested in finding the network topology is the sink.

VI.3 SIMULATION RESULTS

Using the OPNET simulation tool, we implemented our 2hop-clustered NTDP as well as a regular link-state NTDP that uses MPR as is the case for OLSR. We experimented with networks of size 10, 20, 30, 40 and 50 nodes with various values for the transmission range of the wireless nodes and uniformly distributed random placement of nodes over a 100x100 area. In each case, we compared similar topologies for the two protocols and

considered only those topologies that were fully connected and resulted in complete topology discovery by an interested node. Each data point on the graphs is the result of averaging over 5 different random experiments.

Table VI-2 A partial topology table

node	neighbors
v_1	$u_{11}, u_{12}, u_{13}, u_{14}$
v_2	u_{21}, u_{22}, u_{23}
u_{11}	v_1, v_2, u_{14}
u_{12}	v_1
u_{13}	v_1, u_{14}
u_{14}	$v_1, u_{11}, u_{13}, u_{21}$
u_{21}	v_2, u_{14}
u_{22}	v_2, u_{23}
u_{23}	v_2, u_{22}

Figure VI-6 compares the average topology discovery overhead of the two methods per node and normalized to the size of an ID. In the case of the regular link state NTDP, this overhead consists of only topology updates while in the case of 2hop-clustered NTDP it also includes the overhead of JOIN messages, however negligible. In this figure, LS and 2hC stand for Link State and 2hop-Clustered, respectively. As can be seen, in all

situations, the control overhead of our protocol is almost 50% less than its counterpart. Of course, this depends on the counteraction between the number of nodes and the transmission range in terms of the number of CHs that the 2hop-clustered NTDP selects and the percentage of the non-gateway inter-cluster links. The number of CHs influences the number of topology update messages that are generated on one hand and the size of these updates on the other. The smaller the number of CHs, the fewer these messages but the bigger each on average.

The increase in the overhead on the lower side of the transmission range in most cases may be explained by the fact that as the transmission range grows the number of neighbors of each node grows which means the size of update packets would be bigger. On the other hand, the following gradual decrease in the overhead may be attributed to the impact of MPR. In a topology discovery protocol that does not use MPR, the overhead would increase steadily with transmission range because all of the nodes in a fully connected network would repeat each topology update packet exactly once (regardless of the network size). In the case of regular link state protocols, even clustering has no effect because every single node generates a packet. When MPR is used, the number of nodes that repeat each flooded packet decreases with the transmission range which counteracts with the increase in the size of the packet.

Although for any given number of nodes the overhead of topology updates is the smallest when the transmission range is small but we have to warn the reader that for too small transmission ranges the network is often partitioned i.e. it does not form a fully connected graph but is made up of several separate sections. In fact, in our simulations, in

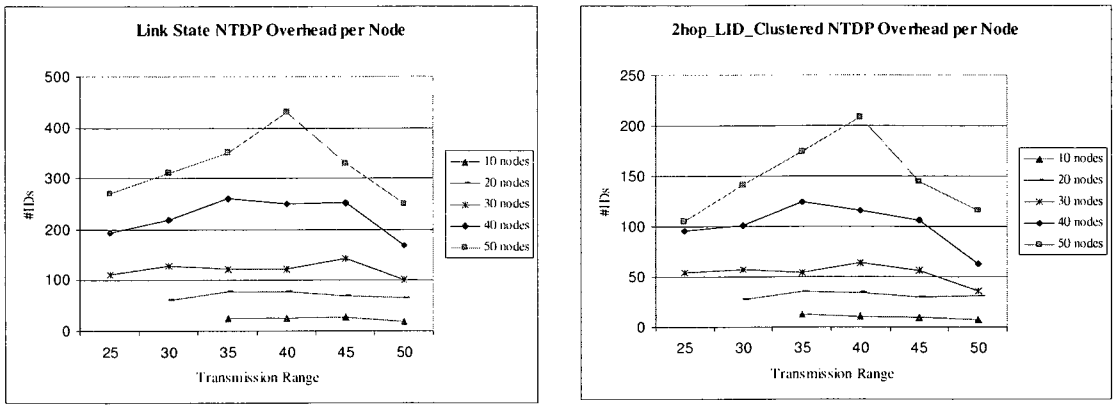


Figure VI-6 Topology discovery OH for link-state and 2hop-clustered schemes

each experiment with a small transmission range we had to try many node distribution alternatives in order to find a fully connected network.

A question that may also come to mind is why per node overhead increases with the number of nodes in the network. This phenomenon can be explained intuitively by realizing that the amount of repeated traffic is not linearly proportional to the number of nodes but exponentially. If n nodes generate T units of overhead in a flooding process, then 2 such groups of nodes mixed in an area, generate much more traffic than $2T$ since each packet from one group would be repeated by some or all of the nodes in the other group as well, depending on the protocol. In other words, the bandwidth consumed per node increases as $O(n)$. Hierarchical architectures are usually used to deal with this issue e.g. the concepts of areas in OSPF, zones in ZRP [27], aggregation points in sensor networks and several approaches mentioned in [133].

VI.4 SUMMARY

In this chapter, we strived to reduce the overhead of periodic routing advertisements in link state routing protocols in order to make them viable for employing in wireless ad hoc networks specially sensor networks. Using clustering, we have reduced the number and the size of these messages. We also found an upper bound on the total overhead under certain conditions. Our simulations have shown a great degree of improvement over a regular link state routing protocol.

Another avenue for overhead reduction that is worth exploring for our protocol is what we refer to as *delta topology updates*, which means advertising only the changes in the neighborhood information at every update interval instead of the whole neighborhood information. This is similar to the differential updates in TBRPF protocol that advertises only the changes in the links state rather than the full current state.

CHAPTER VII

CONCLUSIONS AND FUTURE WORK

In this thesis, we focused on the need of wireless ad hoc routing protocols to provide location privacy and anonymity to users of multihop wireless ad hoc networks. After an extensive survey of the existing literature, we identified several desirable and important characteristics that such a protocol must posses including:

- Be destination-oriented: This property ensures that the destination controls the routing operations and is able to hide information about its identity and location from un-trusted entities as much as possible.
- Be proactive: To eliminate the slow start problem associated with reactive routing protocols and to facilitate destination oriented routing.
- Respect the limitations of wireless devices and networks. For example, in the case of sensor networks, our design uses only symmetric cryptography and relieves sensor nodes from performing computationally intensive routing calculations as well.
- Avoid insecure routing methods. For instance, in sensor networks, sink-originated beacons must not be used.
- Consider the existing trust model. For example, different versions of V-routing must be designed for MANET, community networks and WISP networks.

- Consider the existing traffic model. For example, DCARPS works well in a sensor network with a high volume of traffic while probabilistic DCARPS must be used in a sensor network with few data flows.

Based on these observations, appropriate anonymous routing protocols were proposed for several different environments. We designed three flavors of the V-routing protocol based on a similar framework for MANET and wireless mesh ad hoc networks of type community networks and WISP (Wirelss Internet Service Provider). All of these protocols provide for communication anonymity and sender/receiver location privacy as well as sender/receiver anonymity against a global eavesdropper if sufficient flow diversity exists. In V-routing, the destination node pre-establishes an anonymous virtual hop-by-hop circuit from an arbitrary router in the network towards itself. A future potential sender, will have to send its packets to that router first which will then forward them to the destination node on the pre-established path. We showed how V-routing can be used to support anonymous multicast applications. We discussed the level of security that V-routing offers and its performance in terms of computational and communication overhead.

For Multihop wireless sensor networks, we proposed the Destination Controlled Anonymous Routing Protocol (DCARPS) which supports location privacy for the source nodes and the sink as well as communication anonymity and source/sink anonymity in presence of a global eavesdropper as long as sufficient volume of traffic is available. We evaluated the security performance of this protocol both from a stochastic point of view and using a novel entropy-based metric and simulations. A sink-anonymous topology

discovery protocol was proposed for DCARPS and evaluated using simulations. This protocol was further extended to an all-anonymous topology discovery protocol in order to also resist topology disclosure during topology discovery. As part of the route calculation phase of DCARPS, a load-balancing routing tree algorithm was proposed. Simulations show a great performance improvement compared to a greedy shortest-paths routing algorithm.

Using a stochastic approach, we evaluated the privacy effectiveness of DCARPS. We also used a novel information-theoretic approach in our simulations in order to quantify this concept. This method can be applied to almost any randomized routing protocol.

For wireless sensor networks with low traffic volume, we proposed two probabilistic DCARPS protocols as extensions of the basic DCARPS. In these protocols, the fixed-path routing nature of DCARPS has been replaced by randomized packet forwarding. These protocols support location privacy for the source and the sink as well as sender/sink anonymity and communication anonymity in presence of a local eavesdropper. Using simulations, we compared the source safety period for randomized DCARPS vs. fixed-path routing and flooding.

Both families of V-routing and DCARPS are based on our Destination Controlled Routing (DCR) philosophy. This new approach to routing is the result of adopting the fist and the second principles mentioned at the beginning of this chapter, namely destination-oriented and proactive routing.

In order to accommodate the proactive routing nature of our protocols in a resource-constrained wireless ad hoc network, we introduced the 2hop-clustered network

topology discovery protocol which reduces the overhead of periodic routing updates of regular link state protocols. Using, a redundancy reduction mechanism and the clustering technique, our protocol is able to greatly reduce this overhead. Our simulations show over 50% improvement.

Future Work

- *Extend DCR:* The concept of DCR can be applied to other wireless networks technologies that may exist or may emerge in future. We have used this concept for those network types shown in figure I-3.
- *Mobility:* We have not considered the impact of mobility on the performance of V-routing and DCARPS protocols. In fcat, mobility is rarely dealt with in the context of wireless sensor networks. Obviously, movement of nodes causes the network topology to change and as a result a new round of topology discovery becomes necessary. In V-routing, most of the topology changes are handled by the underlying routing protocol. However, changes at the V-routing sublayer must be handled by this protocol. In general, appropriate maintenance protocols for this protocol must be developed. For example, if an RP or a VHR leaves the network, the corresponding destination nodes must repair their 2^{nd}-leg. Also, when the access router of the destination node changes, the 2^{nd}-leg must be repaired in a secure manner so that the location of this node is not revealed.
- *Scalability:* V-routing has been designed for a small-scale ad hoc network. In a large network, the path establishment part of this protocol may not be practical. Specifically, the correspondence of a node and its RP(s) must be known to all other

nodes or at least access routers. A solution for this problem may be a hierarchical architecture. Also, further improvements with regards to the proactive topology discovery protocols is desirable in order to reduce the control overhead even more.

In DCARPS, the length of the path setup messages depends on the number of nodes and more specifically on the radius of the network. Considering the usually small permissible packet sizes in sensor networks, these messages may have to be segmented into several smaller messages. However, because every such message must still carry the DI labels of the nodes on the path, the scalability problem will still exist although less severe. A solution may be to add a binary flag to each setup packet indicating whether or not it is the last segment of the same message. In that case, each segment carries only the DI labels of the nodes whose routing info it includes. If a node has not received its routing info yet, it examines the outermost DI label in the packet and tries to decrypt the messages. Otherwise, it only rebroadcasts the message. With this protocol, a technique must be invented to prevent flooding of a path setup message out of its own branch and into the whole network since the previously set up nodes cannot distinguish between messages from their own and their neighboring branches.

- *QoS support:* The triangular path in V-routing increases data transmission delay. Obviously, this delay depends on the selection of the RP and the VHRs. However, QoS cannot be easily incorporated as a factor into this selection process because it jeopardizes our location privacy requirement. For example, choosing a shorter 2^{nd}-leg for delay sensitive applications may indicate to the adversary that the destination node is located close to the RP.

On the other hand, QoS-enabled routing in DCARPS is easier. During the topology discovery phase, the sink can collect QoS parameters such as link quality and energy levels from sensors and then use that information during the route calculation phase.

In probabilistic DCARPS, randomized routing causes out-of-order packet delivery and delay jitter. The former problem can be handled at the higher layers of the protocol stack but the latter may be of importance to real-time applications. While jitter removal techniques such as data buffering can be used to solve this problem, we may be able to limit jitter by limiting route variations. Ultimately, this may be a case of security vs. QoS.

- *Standards compatibility:* When designing a new protocol, one must ensure its compatibility with other commonly used protocols. For example, Firewalls and the Network Address Translation (NAT) protocol allow only their local addresses to be used as the source address. This does not create a big problem for V-routing because it only requires the destination identifier to be location-independent. However, it requires the source node to identify itself to the destination node with a different address that it obtains from its NAT server. As another example, in TCP the one-way delay of a connection is important in congestion control. Usually, one-way delay is calculated by measuring the round-trip delay and then dividing that by 2. The underlying assumption in this case is that the one-way delay in both the forward and the reverse directions are equal. However, in most cases this is not true. In particular, in V-routing this is the case since the forward and the reverse connections are most of the time different. One must also make sure that the design for anonymous

communication can safely integrate with other essential elements of a network such as accounting, billing, etc.

- *Hierarchical DCARPS:* In wireless sensor networks, it is common to use in-network processing in order to remove redundancy from the collected data. In this technique, similar to clustering, nodes are organized into local groups and in each group one node takes on the role of an aggregator which collects data from its neighbors and after some processing forwards a reduced amount of data to the sink. In fact, it acts as a local sink. This process may be repeated at several levels of aggregation giving rise to a hierarchy of aggregators. However, because aggregators at one layer normally talk directly to the aggregators at a higher layer, they must use long range transmission which differentiates them from ordinary sensors. This is a security risk similar to the sink-originated beacons. DCARPS can be extended to support location privacy and anonymity for aggregators. In that case, multihopping must be used among aggregators using intermediate ordinary sensors. One of the challenges in this method is establishing the hierarchy, and the sink as the top-layer aggregator, in a secure manner so that the aggregators are not identified but are able to coordinate their activities. This requires a distributed post-deployment pair-wise secret key distribution mechanism.

Aggregation can also help resolve the challenging problem of traffic concentration around the sink. It is a well-known problem that the location of the sink can be identified using physical-layer measurement of the traffic in-flux. Aggregators can reduce this in-flux by removing redundancy and hencefore reduction in the number of forwarded packets as well as rate control. However, the challenge is to

distribute aggregators most effectively and make their traffic incoming pattern as similar to the sink's as possible.

- *DCARPS for multiuser multi-purpose WSN:* Currently, WSNs are highly application-specific in every aspect. However, the research community is moving in the direction of generic multi-purpose sensor networks. This means developing a physical sensor network that can be used by multiple users for various purposes. In terms of security, this means the need for distributed dynamic and anonymous in-the-field pair-wise secret key agreement protocols that can assign shared keys at least to a user-owned sink node and the sensors in the field after deployment as opposed to the pre-deployment initialization.
- *Application dependence:* As we mentioned above, WSNs are application specific. In the design of DCARPS, we only categorized networks based on their traffic volume. But, we believe, that location privacy and anonymity aspects of the network must be investigated for each specific application and according to the capabilities of the sensor devices.
- *Enhancing DCARPS for scarce traffic-sensor networks:* As pointed out in chapter V, our work on probabilistic DCARPS can be improved in terms of scalability, overhead and performance evaluation.
- *Further design enhancements:* Our proposed protocols may have room for improvement. For example, the current design of V-routing may not support communication anonymity with regards to all third parties. In other words, if the source node does not have the storage capability to maintain the information about other nodes' rendezvous points, it must obtain that information from another entity

perhaps its access router which means it has to reveal the identity of the destination to that entity.

Another example is topology disclosure protection for V-routing. This protocol may be combined with other techniques in order to prevent network topology from being revealed to clients or even their access routers

- *Enhancing 2hop-clustered NTDP:*
    - Find the dominating set that generates the least overhead.
    - How quickly should the sink react to a change in topology? It may be temporary, for example due to link congestion.
    - An experimental study is desirable to find out how often, under various conditions, topology changes in a wireless sensor network. Important factors are channel characteristics, transmitter/receiver radios thermal noise fluctuations, modulation and coding types, interference and weather. 2hop-clustered NTDP can be used to periodically obtain topology information in order to keep track of its changes.

REFERENCES

[1] David Chaum, ***Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms***, in the Communications of the ACM 24(2), February 1981, pp. 84-88.

[2] L, Miller, ***No Solitude in Cyberspace,*** USA Today, June 9, 1997.

[3] Andreas Pfitzmann and Michael Waidner, ***Networks without User Observability -- Design Options***, in the proceedings of EUROCRYPT 1985, Springer-Verlag, LNCS 219, 1985, pp. 245-253.

[4] Hughes, D. and Shmatilov, V., ***Information Hiding, Anonymity and Privacy: A Modular Approach***, Journal of Computer Security, 2004, Vol. 12, Issue 1, pp. 3-36.

[5] R. Want, A. Hopper, V. Falcao, J. Gibbons, ***The Active Badge Location System,*** in the ACM Transactions on Information Systems, 1992, 10(1), pp. 91-102.

[6] Roy Want, Bill N. Schilit, Norman I. Adams, Rich Gold, Karin Petersen, David Goldberg, John R. Ellis and Mark Weiser, ***An Overview of the PARCTAB Ubiquitous Computing Experiment***, IEEE Personal Communications, December 1995, Vol.2, No.6, pp. 28-43.

[7] ***FCC 911 Order***, www.fcc.gov/Bureaus/Wireless/Orders/fcc96264.txt. (visited on Nov. 8, 2008)

[8] Baik Hoh, Marco Gruteser, ***Protecting Location Privacy Through Path Confusion***, in IEEE/CreateNet International. Conference on Security and Privacy for Emerging Areas in Communication Networks (SecureComm), Athens, Greece, September 5-9, 2005, pp. 194-205.

[9] Alastair R. Beresford and Frank Stajano, ***Location Privacy in Pervasive Computing***, IEEE Pervasive Computing, January-March 2003, Vol.2, Issue 1, pp. 46-55.

[10] ***Location Privacy Protection Act***, http://www.techlawjournal.com/cong107/privacy/ location/s1164is.asp, 2001. (visited on Nov. 8, 2008)

[11] Charles E. Perkins and P. Bhagwat, ***Routing over Multihop Wireless Network of Mobile Computers,*** in Mobile Computing, edited by Tomasz Imielinski and Henry F. Korth, Kluwer Adademic Publishers, 1996, Chapter 6, pp. 183-206.

[12] S. Murthy and J.J. Garcia-Luna-Aceves, ***An Efficient Routing Protocol for Wireless Networks***, ACM/Baltzer Journal on Mobile Networks and Applications, Special Issue on Routing in Mobile Communication Networks, Vol. 1, No. 2, October 1996, pp. 183-197.

[13] Tsu-Wei Chen and Mario Gerla, ***Global State Routing: A New Routing Scheme for Ad-hoc Wireless Networks***, in the proceedings of the IEEE ICC'98, June 1998, pp. 171-175.

[14] Iwata, C.-C. Chiang, G. Pei, M. Gerla, and T.-W. Chen, ***Scalable Routing Strategies for Ad Hoc Wireless Networks***, IEEE Journal on Selected Areas in Communications, Special Issue on Ad-Hoc Networks, August 1999, pp.1369-79.

[15] M Joa-Ng and I.-T. Lu, ***A Peer-to-Peer Zone-based Two-level Link State Routing for Mobile Ad Hoc Networks,*** IEEE Journal on Selected Areas in Communications, Special Issue on Ad-Hoc Networks, August 1999, pp. 1415-1425.

[16] C.-C. Chiang, ***Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel***, in the Proceedings of the IEEE Singapore International Conference on Networks (SICON'97), April 1997, pp. 197-211.

[17] David B. Johnson, David A. Maltz, and Josh Broch, ***DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks,*** in Ad Hoc Networking, edited by Charles E. Perkins, Addison-Wesley, 2001, Chapter 5, pp. 139-172.

[18] Johnson, Maltz, Hu, ***The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)***, IETF Internet draft, 19-July-04, http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-10.txt. (visited on Nov. 8, 2008)

[19] Young-Bae Ko and Nitin Vaidya, ***Location-Aided Routing (LAR) in Mobile Ad Hoc Networks***, in the proceedings of the Fourth International Conference on Mobile Computing and Networking (MobiCom'98), ACM, October 1998, pp. 66–75.

[20] Charles E. Perkins and Elizabeth M. Royer, ***Ad hoc On-Demand Distance Vector Routing***, in the proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA, February 1999, pp. 90-100.

[21] Charles E. Perkins, Elizabeth M. Belding-Royer, and Samir Das, ***Ad Hoc On Demand Distance Vector (AODV) Routing***, IETF RFC 3561, http://www.faqs.org/rfcs/rfc3561.html. (visited on Nov. 8, 2008)

[22] Mingliang Jiang, Jinyang Li, Y.C. Tay, ***Cluster Based Routing Protocol***, IETF Internet Draft, August 1999, http://www.math.nus.edu.sg/~mattyc/cbrp.txt. (visited on Nov. 8, 2008)

[23] V.D Park and M.S Corson, ***A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks***, in the proceedings of the INFOCOM'97, April 1997, Vol.3, pp. 1405 - 1413.

[24] Chai-Keong Toh, ***A Novel Distributed Routing Protocol to Support Ad hoc Mobile Computing***, in the proceedings of IEEE 15th Annual International Phoenix Conference On Computation and Communication, March 1996, pp. 480-486.

[25] R. Dube et al, ***Signal Stability Based Adaptive Routing for Ad Hoc Mobile Networks***, IEEE Personal Communications, February 1997, pp. 36-45.

[26] Rajendra V. Boppana, Satyadeva P Konduru, ***An Adaptive Distance Vector Routing Algorithm for Mobile, Ad Hoc Networks,*** in the proceedings of the IEEE INFOCOM 2001, pp. 1753-1762.

[27] Zygmunt J. Haas, Marc R. Pearlman, Prince Samar, ***The Zone Routing Protocol (ZRP) for Ad hoc Networks,*** IETF, MANET Internet draft, July 2002, http://tools.ietf.org/id/draft-ietf-manet-zone-zrp-04.txt. (visited on Nov. 8, 2008)

[28] T. Clausen, P. Jacquet, A. Laoiti, P. Minet, P. Muhlethaler, A. Qayyum, L. Viennot, ***Optimized Link State Routing Protocol,*** IETF RFC 3626, October 2003, http://www.ietf.org/rfc/rfc3626.txt. (visited on Nov. 8, 2008)

[29] R. Ogier, F. Templin, M. Lewis, ***Topology Dissemination Based on Reverse-Path Forwarding (TBRPF)***, IETF RFC 3684, February 2004, http://www.ietf.org/rfc/rfc3684.txt. (visied on Nov. 8, 2008)

[30] J.J. Garcia-Luna-Aceves, Marcelo Spohn, ***Source-Tree Routing in Wireless Networks,*** in the proceedings of the seventh International Conference on Network Protocols (ICNP99), Toronto, Canada, October 31-November 3, 1999, page 273.

[31] R. Ramanathan and M. Steenstrup, ***Hierarchically-Organized, Multihop Mobile Networks for Multimedia Support***, ACM/Baltzer Mobile Networks and Applications, , 1998, Vol. 1, No. 2, pp. 89-103.

[32] Rodrigo Fonseca, Omprakash Gnawali, Kyle Jamieson, Sukun Kim, Philip Levis, and Alec Woo, ***The Collection Tree Protocol (CTP)***, August 2006, http://www.tinyos.net/tinyos-2.x/doc/txt/tep123.txt. (visited on Nov. 8, 2008)

[33] Samuel Madden, Michael J. Franklin, Joseph Hellerstein, and Wei Hong, ***TAG: a Tiny Aggregation Service for Ad-Hoc Sensor Networks***, in the proceedings of the Fifth Symposium on Operating Systems Design and implementation (OSDI '02), December 9 - 11, 2002, Boston, MA, USA, pp. 131-146.

[34] Chalermek Intanagonwiwat, Ramesh Govindan, Deborah, ***Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks***, in the proceedings of the 6th annual international conference on mobile computing and networking (MobiCom2000), August 2000, Boston, MA, USA, pp. 56-67.

[35] Fan Ye, Alvin Chen, Songwu Lu, Lixia Zhang, ***A Scalable Solution to Minimum Cost Forwarding in Large Sensor Networks,*** in the proceedings of the 10th International Conference on Computer Communications and Networks, 2001, pp. 304–309.

[36] David Braginsky, Deborah Estrin, ***Rumor Routing Algorithm For Sensor Networks***, in the proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications, 2002, Atlanta, Georgia, USA, pp. 22-31.

[37] Eran Gabber, Phillip B. Gibbons, Yossi Matias, and Alain Mayer, ***How to Make Personalized Web Browsing Simple, Secure, and Anonymous***, in the proceedings of Financial Cryptography '97, LNCS 1318, pp. 17-31.

[38] M. Reed, P. Syverson, and D. Goldschlag, ***Proxies for Anonymous Routing***, in the proceedings of the 12th Annual Computer Security Applications Conference, San Diego, CA, December 1995, pp. 95-104.

[39] Shu Jiang, N.H Vaidya, Wei Zhao, ***A Mix Route Algorithm for Mix-net in Wireless Mobile Ad hoc Networks,*** in the proceedings of the IEEE International Conference on Mobile Ad-hoc and Sensor Systems, October 2004, pp. 406 – 415.

[40] El-Khatib, K., Korba, L, Song, R., and Yee, G., ***Secure Dynamic Distributed Routing Algorithm for Ad Hoc Wireless Networks,*** in the proceedings of the 1st International Workshop on Wireless Security and Privacy (WiSPr 2003), Kaohsiung, Taiwan. October 2003, page 359.

[41] Jiejun Kong, Xiaoyan Hong, Mario Gerla, ***An Anonymous On Demand Routing Protocol with Untraceable Routes for Mobile Ad-hoc Networks,*** in the proceedings of the 4th ACM International Symposium on Mobile Ad hoc Networking and Computing (*MobiHoc'03),* June 2003, Annapolis, Maryland, USA, pp. 291-302 .

[42] Y. Zhang, W. Liu, and W. Lou, ***Anonymous Communications in Mobile Ad Hoc Networks***, in the proceedings of the IEEE INFOCOM, Miami, FL, March 2005, Vol.3, pp. 1940- 1951.

[43] P.F. Syverson, D.M. Goldschlag, and M.G. Reed, ***Anonymous Connections and Onion Routing,*** in the proceedings of the IEEE Symposium on Security and Privacy, Oakland, California, 1997, pp. 44-54.

[44] D. Samfat, R. Molva, and N. Asokan, ***Untraceability in Mobile Networks,*** in the proceedings of the 1st Annual International Conference on Mobile Computing and Networking (*MOBICOM),* December 1995, pp. 26–36.

[45] R. Molva, D. Samfat, and G. Tsudik, ***Authentication of Mobile Users,*** IEEE Network, 8(2), 1994, pp. 26–34.

[46] Yih-Chun Hu, David B. Johnson, Adrian Perrig, ***SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks***, in the proceedings of the 4th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'02), June 2002, page 3.

[47] Yih-Chun Hu, Adrian Perrig, David B. Johnson, ***Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks***, *MobiCom'02,* September 23–26, 2002, Atlanta, Georgia, USA, pp. 12-23.

[48] Panagiotis Papadimitratos and Zygmunt J. Haas, ***Secure Routing for Mobile Ad hoc Networks,*** in the proceedings of the SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002), San Antonio, TX, January 27-31, 2002.

[49] Bridget Dahill, Kimaya Sanzgiri, Brian N.Levine, Elizabeth M.Belding-Royer, Clay Shields, ***A Secure Routing Protocol for Ad Hoc Networks,*** in the proceedings of the 10th IEEE International Conference on Network Protocols, November 2002, pp. 78–87.

[50] M.G. Zapata, ***Secure Ad hoc On-Demand Distance Vector (SAODV) Routing,*** IETF Internet draft, August 2002, draft-guerrero-manet-saodv-00.txt. (visited on Nov. 8, 2008)

[51] Manel Guerrero Zapata and N. Asokan, ***Securing Ad Hoc Routing Protocols***, in the proceedings of the first ACM Workshop on Wireless Security (WiSe 2002), September 2002 Atlanta, GA, USA, pp. 1-10.

[52] Seung Yi, Prasad Naldurg, Robin Kravets, ***Security-Aware Ad Hoc Routing for Wireless Networkss***, in the proceedings of the 2nd ACM International Symposium on Mobile Ad hoc Networking & Computing, Long Beach, CA, USA, 2001, pp. 299 - 302.

[53] L. Zhou and Z. J. Haas, ***Securing Ad Hoc Networks,*** IEEE Network Magazine, November 1999, pp. 24-30.

[54] B. Smith, S. Murthy and J.J. Garcia-Luna-Aceves, ***Securing Distance Vector Routing Protocols,*** in the Internet Society Symposium on Network and Distributed System Security, the 7th International Workshop on Security Protocols, San Diego, CA, USA, February 1997, pp. 85-92.

[55] Ronggong Song, Larry Korba, George Yee, ***AnonDSR: Efficient Anonymous Dynamic Source Routing for Mobile Ad-Hoc Networks***, in the proceedings of the

ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN 2005), Alexandria, Virginia, USA, November 7-11, 2005, pp. 32-42.

[56] J. Al-Muhtadi, R. Campbell, A. Kapadia, M. Dennis Mickunas, Seung Yi, ***Routing Through the Mist: Privacy Preserving Communication in Ubiquitous Computing Environments,*** in the proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS'02), July 2002, Vienna, Austria, page 74.

[57] Azzedine Boukerche, Khalil El-Khatib, Li Xu, Larry Korba, ***A Novel Solution for Achieving Anonymity in Wireless Ad hoc Networks***, PE-WASUN'04, October 7, 2004, Venizia, Italy, pp. 30-38.

[58] Xiaoxin Wu and Bharat Bhargava, ***AO2P: Ad Hoc On-Demand Position-Based Private Routing Protocol,*** IEEE Transactions on Mobile Computing, Vol.4, No.4, July/August 2005, pp. 335-348.

[59] Srdjan Capkun, Jean-Pierre Hubaux and Markus Jakobsson. ***Secure and Privacy-Preserving Communication in Hybrid Ad Hoc Networks,*** EPFL-IC Technical Report IC/2004/10, January 2004.

[60] Jalel Ben-Othman, Xiaoyun Xue, ***SERAN: A new protocol to hide an equipment in ad hoc networks,*** in the proceedings of the 8th IEEE International Symposium on Computers and Communication (ISCC'03), Vol.1, pp. 356-361.

[61] B. Zhu, Z. Wan, M. Kankanhalli, F. Bao, and R. Deng, ***Anonymous Secure Routing in Mobile Ad-hoc Networks,*** in the proceedings of the 29th Annual IEEE International Conference on Local Computer Networks (LCN), 2004, pp. 102–8.

[62] S. Seys and B. Preneel, ***ARM: Anonymous Routing Protocol for Mobile Ad hoc Networks***, in the proceedings of the the 20th International Conference on Advanced Information Networking and Applications (AINA), Vol. 2, April 2006, pp. 133–137.

[63] Karim El Defrawy and Gene Tsudik, ***PRISM: Privacy-friendly Routing In Suspicious MANETs (and VANETs),*** in the proceedings of the 2008 IEEE International Conference of Network Protocols (ICNP'08), October 19-22, Florida, USA, pp. 258-267.

[64] Jung-Chun Kao and Radu Marculescu, ***Real-Time Anonymous Routing for Mobile Ad Hoc Networks,*** in the proceedings of the WCNC, Kowloon, Hong Kong, March 11-15, 2007, pp. 4139-4144.

[65] Jun Liu, Xiaoyan Hong, Jiejun Kong, Qunwei Zheng, Ning Hu, Phillip G. Bradford, ***A Hierarchical Anonymous Routing Scheme for Mobile Ad-Hoc Networks,*** in the proceedings of Milcom06, Washington D.C., October 2006, pp.1-7.(reduces public key operations using a hierarchy)

[66] S. Jiang, N. H. Vaidya, and W. Zhao, ***A mix route algorithm for mixnet in wireless ad hoc networks***, in the Proceedings of the IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS), October 2004, pp. 24- 27. (adapts mix net to dynamic ad hoc topologies)

[67] Y.-C. Hu and H. J. Wang, ***A Framework for Location Privacy in Wireless Networks***, in the proceedings of the ACM SIGCOMM Asia Workshop, Beijing, China, April 2005. (employs a callback service and requires a secure out-of-band channel)

[68] M. J. L. Yang and S. Wetzel, ***Discount anonymous on demand routing for mobile ad hoc networks***, in the proceedings of SECURECOMM2006, Baltimore, MD, USA, pp. 1-10 (similar to Onion but without the use of PKI)

[69] D. Sy, R. Chen, and L. Bao, ***ODAR: On-Demand Anonymous Routing in Ad Hoc Networks***, in the proceedings of the 3rd IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS), 2006, Vancouver, Canada, pp. 267-276.

[70] Lichun Bao, ***A New Approach to Anonymous Multicast Routing in Ad Hoc Networks***, in the proceedings of the 2nd International Conference on Communications and Networking (CHINACOM), Shanghai, China, August 21-24, 2007, pp. 1004-1008.

[71] L. Xiao, Y. Liu, W. Gu, D. Xuan, and X. Liu, ***Mutual anonymous overlay multicast***, Journal of Parallel Distributed Computing, 66(9), 2006, pp. 1205–16.

[72] Jung-Chun Kao and Radu Marculescu, ***Energy-Efficient Anonymous Multicast in Mobile Ad-Hoc Networks,*** in the proceedings of the 13th International Conference on Parallel and Distributed Systems (ICPADS), Hsinchu, Taiwan, Dec. 2007, Vol.1, pp.1-8.

[73] Celal Ozturk, Yanyong Zhang, Wade Trappe, Source-Location Privacy in Energy-Constrained Sensor Network Routing, in the proceedings of the ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN'04), October 2004, Washington, DC, pp. 88-93.

[74] Pandurang Kamat, Yanyong Zhang, Wade Trappe, Celal Ozturk, ***Enhancing Source-Location Privacy in Sensor Network Routing***, in the proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICSCS'05), Vol. 00, pp. 599 – 608.

[75] Y. Xi, L. Schwiebert, and W. Shi, ***Preserving Privacy in Monitoring-based Wireless Sensor Networks,*** in the proceedings of the 2nd International Workshop on Security in Systems and Networks (SSN), April 2006, Rhodes Island, Greece.

[76] Y. Ouyang, Z. Le, G. Chen, J. Ford, and F. Makedon, ***Entrapping adversaries for source protection in sensor networks***, in the proceedings of the International Symposium on World of Wireless, Mobile and Multimedia Networks (WoWMoM), June 2006, pp. 23–34.

[77] Kiran Mehta, ***Location Privacy in Sensor Networks Against a Global Eavesdropper,*** in the proceedings of the IEEE International Conference on Network Protocols (ICNP), October 2007, Beijing, China, pp. 314-323.

[78] Xiaoyan Hong, Pu Wang, Jiejun Kong, Qunwei Zheng, Jun Liu, ***Effective Probabilistic Approach Protecting Sensor Traffic***, in the proceedings of the IEEE Military Communication Conference (Milcom) 2005, Atlantic City, NJ, October 2005, Vol.1, pp. 169-175.

[79] Shu Jiang, Nitin H.Vaidya, Wei Zhao, ***Routing in Packet Radio Networks to Prevent Traffic Analysis***, in the proceedings of the IEEE Information Assurance and Security Workshop, West Point, NY, July 2000.

[80] Jing Deng, Richard Han, Shivakant Mishra, ***Intrusion Tolerance and Anti-Traffic Analysis Strategies For Wireless Sensor Networks***, in the proceedings of the IEEE International Conference on Dependable Systems and Networks (DSN 2004), 28 June - 1 July 2004, Florence, Italy, page 637.

[81] Jing Deng, Richard Han, Shivakant Mishra, ***Countermeasures Against Traffic Analysis Attacks in Wireless Sensor Networks***, 1st International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM'05), Athens, Greece, pp. 113-126.

[82] Stephan Olariu, Mohamed Eltoweissy, Mohamed Younis, ***ANSWER: Autonomous Wireless Sensor Network***, in the proceedings of the Q2SWinet'05, October 13, 2005, Montreal, Quebec, Canada, pp. 88-95.

[83] A. Wadaa, S. Olariu, L. Wilson, M. Eltoweissy, K. Jones, ***On Providing Anonymity in Wireless Sensor Networks,*** in the Proceedings of the Tenth International Conference on Parallel and Distributed Systems (ICPADS'04), Newport Beach, California, July 7-9, 2004, pp. 411-418.

[84] A. Wadda, S. Olariu, L. Wilson, M. Eltoweissy, K. Jones, ***Training a Wireless Sensor Network,*** Journal of Mobile Networks and Applications (MONET), Kluwer Academic Publishers, February 2005, pp. 151-168.

[85] S. Olariu, Q. Xu, M. Eltoweissy, A. Wadaa, A.Y. Zomaya, ***Protecting the Communication Structure in Sensor Networks,*** International Journal of Distributed Sensor Networks, Taylor and Francis, Vol. 1, No. 2, April-June 2005, pp. 187-203.

[86] J. Moy, ***OSPF Version 2***, IETF RFC 2328, April 1998. www.ietf.org/**rfc/rfc2328**.tx (visited on Nov. 8, 2008)

[87] C. Hedrick, ***Routing Information Protocol***, IETF RFC 1058, June 1988, http://www.ietf.org/rfc/rfc1058.txt (visited on Nov. 8, 2008)

[88] ***IEEE 802.11n Status Update***, http://grouper.ieee.org/groups/802/11/Reports/tgn_update.htm (visited on Nov. 8, 2008)

[89] V. Navda, A. Kashyap, S.R. Das, ***Design and Evaluation of iMesh: An Infrastructure-mode Wireless Mesh Network,*** 6th IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks, WoWMoM 2005, 13-16 June 2005, pp. 164- 170.

[90] Krishna N. Ramachandran, Milind M. Buddhikot, Girish Chandranmenon, Scott Miller, Elizabeth M. Belding-Royer, Kevin C. Almeroth, ***On the Design and Implementation of Infrastructure Mesh Networks,*** 1st IEEE Workshop on Wireless Mesh Networks (WiMesh-2005) held in conjunction with SECON-2005, Santa Clara, CA, 26th September, 2005, .

[91] http://www.connect802.com/firetide.htm (visited on Nov. 8, 2008)

[92] http://www.ieee802.org/11/PARs/11-04-0054-02-0mes-par-ieee-802-11-ess-mesh.doc (visited on Nov. 8, 2008)

[93] Takahiro Fujiwara, Noboru Iida, Takashi Watanabe, ***An Ad-hoc Routing Protocol in Hybrid Wireless Networks for Emergency Communications,*** in the proceedings of the 24th International Conference on Distributed Computing Systems Workshops (ICDCSW'04), Vol. 7, pp. 748-754.

[94] Dave Wisely and Enric Mitjana, ***Paving the Road to beyond 3G – the 1st MIND Project***, the 13th IEEE International Symposium on Personal Indoor and Mobile Radio Communications, Vol.3, September 15-18, 2002, pp. 1042-1046.

[95] Hongyi Wu, Chunming Qiao, Swades De, and Ozan Tonguz, ***Integrated Cellular and Ad Hoc Relaying Systems: iCAR,*** IEEE JSAC Vol.19, No. 10, October 2001, pp. 2105-2115.

[96] Srdjan Capkun, Jean-Pierre Hubaux and Markus Jakobsson. ***Secure and Privacy-Preserving Communication in Hybrid Ad Hoc Networks,*** EPFL-IC Technical Report IC/2004/10, January 2004.

[97] Eleonora Borgia, Marco Conti, Franca Delmastro, Luciana Pelusi, ***Lessons from an Ad hoc Network Test-Bed: Middleware and Routing Issues***, Ad Hoc & Sensor Wireless Networks, Old City Publishing, Inc., Vol. 1, pp. 125-157.

[98] C.A. Santivanez, B. McDonald, I. Stavrakakis, R. Ramanathan, ***On the Scalability of Ad hoc Routing Protocols,*** in the Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2002, Vol. 3, Issue 2002, pp. 1688 – 1697.

[99] C. Tschudin, H. Lundgren, H. Gulbrandsen, ***Active Routing for Ad Hoc Networks***, IEEE Communications Magazine, April 2000, pp. 122-127.

[100] ***Information Technology-Telecommunications and Information Exchange between Systems- Local and Metropolitan Area Networks- Specific Requirements- Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications,*** IEEE Std 802.11-1997.

[101] C. Perkins, ***IP Mobility Support***, IETF RFC 2002, October 1996.

[102] Akyildiz, I.F., Xudong Wang, *A Survey on Wireless Mesh Networks,* IEEE Communications Magazine, Volume 43, Issue 9, Sept. 2005, pp. S23-S30.

[103] http://en.wikipedia.org/wiki/Strawberry_Fair (visited on Nov. 8, 2008)

[104] http://www.govtech.com/gt/96814 (visited on Nov. 8, 2008)

[105] http://www.army-technology.com/contractors/navigation/rajant/ (visited on Nov.8, 2008)

[106] http://www.belairnetworks.com/resources/pdfs/WP_Municipal_BDMC00080-A01.pdf (visited on Nov.8, 2008)

[107] http://michaelocc.com/2006/09/downtown-toronto-hydro-wifi-grid.html (visited on Nov. 8, 2008)

[108] http://en.wikipedia.org/wiki/List_of_wireless_community_networks_by_region (visited on Nov. 8, 2008)

[109] http://www.viasyscorp.com/news/latest.htm (visited on Nov. 8, 2008)

[110] http://www.motorola.com/governmentandenterprise/contentdir/de_DE/Files/ProductInformation/MESH_MEA%20Network.pdf (visited on Nov. 8, 2008)

[111] http://www.wi-fiplanet.com/news/article.php/3549846 (visited on Nov. 8, 2008)

[112] http://www.wi-fiplanet.com/news/article.php/3553326 (visited on Nov. 8, 2008)

[113] S. Kent, ***IP Encapsulating Security Payload (ESP)***, IETF RFC 4303, December 2005, http://www.ietf.org/rfc/rfc4303.txt (visited on Nov. 8, 2008)

[114] Andreas Fasbender, Dogan Kesdogan, and Olaf Kubitz, ***Analysis of Security and Privacy in Mobile IP,*** in the 4th International Conference on Telecommunication Systems Modeling & Analysis, Nashville, USA, March 1996.

[115] Adi Shamir, ***How to Share a Secret,*** in the Communications the ACM, No. 11, Vol. 22, November 1979, pp. 612-613.

[116] Y. Desmedt and Y. Frankel, ***Threshold Cryptosystems***, Advances in Cryptology-Crypto '89 (lecture Notes in Computer Science 435), August 1990, pp. 307-315.

[117] Y. Desmedt, ***Threshold Cryptography***, European Transactions on Telecommunications, Vol. 5, No. 4, July-August 1994, pp. 449-457.

[118] Li Xiao, Xiaomei Liu, Wenjun Gu, Dong Xuan, Yunhao Liu, ***A design of overlay anonymous multicast protocol,*** Journal of Parallel and Distributed Computing, Vol. 66 , Issue 9, September 2006, Special issue: Security in grid and distributed systems, pp. 1205 – 1216.

[119] Jamal N. Al-Karaki, Ahmed E. Kamal, ***Routing Techniques in Wireless Sensor Networks: A Survey***, IEEE Wireless Communications, Vol. 11, No. 6. (2004), pp. 6-28.

[120] Wei Ye, John Heidemann and Deborah Estrin, ***An Energy-Efficient MAC Protocol for Wireless Sensor Networks,*** In the proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002), New York, NY, USA, June, 2002, Vol.3, pp. 1567- 1576.

[121] ***Mica Architecture***, http://webs.cs.berkeley.edu/retreat-1-02/MICA_Node_Architecture.ppt (visited on Nov. 8, 2008)

[122] ***Crossbow MICA Mote Product Specifications***, http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA.pdf (visited on Nov. 8, 2008)

[123] Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, ***Wireless Sensor Networks for Habitat Monitoring***, WSNA'02, Atlanta, Georgia, September 28, 2002, pp. 88-97.

[124] B. Deb, S. Bhatnagar, and B. Nath, ***A topology Discovery Algorithm for Sensor Networks with Applications to Network Management***, Department of Computer Science, Rutgers University, Technical Report DCS-TR-441, 2001.

[125] B. Deb, S. Bhatnagar, and B. Nath, ***Multi-resolution State Retrieval in Sensor Networks,*** in the proceedings of the IEEE ICC Workshop on Sensor Network Protocols and Applications, Anchorage, AK, May 2003, pp. 19-29.

[126] Y. Thomas Hou, Yi Shi, Hanif D. Sherali and Scott F. Midkiff, ***Prolonging Sensor Network Lifetime with Energy Provisioning and Relay Node Placement,*** IEEE SECON, Santa Clara, California, USA, September 2005, pp. 295-304.

[127] R. Beresford, ***Location Privacy in Ubiquitous Computing***, Technical Report. Number 612. Computer Laboratory, Cambridge University, UCAM-CL-TR-612. ISSN 1476-2986, January 2005.

[128] Alberto Leon-Garcia, Indra Widjaja, ***Communication Networks***, McGraw Hill, 2000, pp. 497-499.

[129] Prasanth Ganesan, Ramnath Venugopalan, Pushkin Peddabachagari, Alexander Dean, Frank Mueller, Mihail Sichitiu, ***Analyzing and Modeling Encryption Overhead for Sensor Network Nodes***, in the proceedings of the 2nd ACM international conference on Wireless sensor networks and applications (WSNA'03), September 2003, San Diego, USA, pp. 151 – 159.

[130] Y.W. Law, J. Doumen, P. Hartel, ***Benchmarking Block Ciphers for Wireless Sensor Networks***, in the proceedings of the IEEE International Conference on Mobile Ad-hoc and Sensor Systems, 25-27 October 2004, pp. 447- 456.

[131] Yangcheng Huang, Saleem Bhatti, Søren-Aksel Sørensen, ***Adaptive MANET Routing for Low Overhead,*** in the proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM 2007, Espoo, Finland, pp 1-6.

[132] Qayyum, L. Viennot, A.Laouiti, ***Multipoint Relaying for Flooding Broadcast Messages in Mobile Wireless Networks***, in the proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02), Vol. 9, January 2002, pp. 298-308.

[133] Cesar A. Santivanez, Ioannis Stavrakakis, Ram Ramanathan, ***Making LinkState Routing Scale for Ad Hoc Networks***, in the proceedings of MobiHoc 2001, Long Beach, CA, USA, October, 2001, pp. 22-32.

[134] M. Gerla and T.C. Tsai, ***Multicluster, Mobile, Multimedia Radio Network***, Wireless Networks, Vol.1, 1995, pp. 255–265.

[135] Stefano Basagni, ***Finding a Maximal Weighted Independent Set in Wireless Networks***, Telecomm. Systems, 18:1–3, 2001, pp. 155–168.

[136] S. Basagni, I. Chlamtac and A. Farago, ***A Generalized Clustering Algorithm for Peer-to-Peer Network***, Workshop on Algorithmic Aspects of Communication (satellite workshop of ICALP), Bologna, Italy, July 1997.

APPENDIX A

RFC 4122

REC 4122 defines a Uniform Resource Name (URN) namespace for UUIDs (Universally Unique IDentifier), also known as GUIDs (Globally Unique IDentifier). A UUID is an identifier that is unique across both space and time with respect to the space of all UUIDs. A UUID has a fixed size of 128 bits and contains a time field. UUIDs were originally used in the computing system but they are now used for multiple purposes, from tagging objects with an extremely short lifetime, to reliably identifying very persistent objects across a network.

`The UUID generation algorithms support very high allocation rates of up to 10 million per second per device if necessary, so that they could even be used as transaction IDs. One of the main reasons for using UUIDs is that no centralized authority is required to administer them (although one format uses IEEE 802 node identifiers, others do not). As a result, generation on demand can be completely automated, and used for a variety of purposes.

This RFC specifies three algorithms for generating unique UUIDs. The first algorithm leverages the unique values of 802 MAC addresses to guarantee uniqueness, the second uses pseudo-random number generators, and the third uses cryptographic hashing and application-provided text strings. As a result, the generated UUIDs will be unique from all other UUIDs that have been or will be assigned.

The name-based UUID generation algorithm is meant for generating UUIDs from "names" that are drawn from, and unique within, some "name space". The concept of name and name space should be broadly construed, and not limited to textual names. For example, some name spaces are the domain name system, URLs, ISO Object IDs (OIDs), X.500 distinguished Names (DNs), and reserved words in a programming language. The requirements for these types of UUIDs are as follows:

- The UUIDs generated at different times from the same name in the same namespace must be equal.
- The UUIDs generated from two different names in the same namespace should be different (with very high probability).
- The UUIDs generated from the same name in two different namespaces should be different with (very high probability).
- If two UUIDs that were generated from names are equal, then they were generated from the same name in the same namespace (with very high probability).

PUBLICATIONS

[1] Alireza A. Nezhad, Ali Miri, Dimitris Makrakis, ***Location-Concealing and Anonymity-Preserving Routing for Wireless Ad hoc Networks***, in the proceedings of the Canadian Workshop on Information Theory (CWIT), Montreal, Canada, June 2005, pp. 284–287.

[2] Alireza A. Nezhad, Dimitris Makrakis, Ali Miri, ***Location Concealing Probabilistic Routing for Wireless Sensor Networks***, in the proceedings of the 7th IASTED International Conferences on Wireless and Optical Communications, Montreal, Canada, May 30 – June 1, 2007, pp. 87-92.

[3] Alireza A. Nezhad, Ali Miri, Dimitris Makrakis, Luis O. Barbosa, ***Anonymous Proactive Routing for Wireless Infrastructure Mesh Networks***, in the proceedings of the International Conference on Wireless Sensor and Actor Networks (WSAN'07), Spain, Sept.2007, pp. 71-82.

[4] Alireza A. Nezhad, Dimitris Makrakis, Ali Miri, ***Destination Controlled Anonymous Routing in Resource Constrained Multihop Wireless Sensor Networks***, in the proceedings of the 1st International Conference on Wireless Sensor and Actor Networks (WSAN'07), Spain, Sept.2007, pp. 83-94.

[5] Alireza A. Nezhad, Ali Miri, Dimitris Makrakis, Luis O. Barbosa, ***Privacy within Pervasive Communications***, in the special issue of the Springer´s Telecommunications Systems journal, Vol.40, Iss.3 (2009), page 101.

[6] Alireza A. Nezhad, Ali Miri, Dimitris Makrakis, ***Location Privacy and Anonymity Preserving Routing for Wireless Sensor Networks,*** the Elsevier's International Journal of Computer and Telecommunications Networking (Computer Networks), Dec. 2008, Vol.52, Iss.8, pp. 3433-3452.

[7] Alireza A.Nezhad, Dimitris Makrakis, Ali Miri, ***Anonymous Topology Discovery for Wireless Sensor Networks***, in the proceedings of the 3rd ACM International Workshop on QoS and Security for Wireless and Mobile Networks (Q2SWinet'07), Chania, Crete Island, Greece, Oct. 2007, pp. 78-85.

[8] Alireza A. Nezhad, Ali Miri, Dimitris Makrakis, ***Efficient Topology Discovery for Multihop Wireless Sensor Networks***, in the proceedings of the 6th annual conference on Communication Networks and Services Research (CNSR2008), Halifax, Nova Scotia, Canada, May 5 - 8, 2008, pp. 358-365.