STARS

University of Central Florida
STARS

Electronic Theses and Dissertations, 2004-2019

2006

Collaboration Enforcement In Mobile Ad Hoc Networks

Ning Jiang
University of Central Florida

 Part of the Computer Sciences Commons, and the Engineering Commons
Find similar works at: https://stars.library.ucf.edu/etd
University of Central Florida Libraries http://library.ucf.edu

This Doctoral Dissertation (Open Access) is brought to you for free and open access by STARS. It has been accepted for inclusion in Electronic Theses and Dissertations, 2004-2019 by an authorized administrator of STARS. For more information, please contact STARS@ucf.edu.

STARS Citation
Jiang, Ning, "Collaboration Enforcement In Mobile Ad Hoc Networks" (2006). *Electronic Theses and Dissertations, 2004-2019*. 844.
https://stars.library.ucf.edu/etd/844

COLLABORATION ENFORCEMENT IN MOBILE AD HOC NETWORKS

by

NING JIANG
B.S. Shanghai Jiao Tong University 1997
M.S. University of Central Florida, 2001

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the School of Electrical Engineering and Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida
Major Professor: Dr. Kien A. Hua

Spring Term
2006

© 2006 Ning Jiang

ABSTRACT

Mobile Ad hoc NETworks (MANETs) have attracted great research interest in recent years. Among many issues, lack of motivation for participating nodes to collaborate forms a major obstacle to the adoption of MANETs. Many contemporary collaboration enforcement techniques employ reputation mechanisms for nodes to avoid and penalize malicious participants. Reputation information is propagated among participants and updated based on complicated trust relationships to thwart false accusation of benign nodes. The aforementioned strategy suffers from low scalability and is likely to be exploited by adversaries. To address these problems, we first propose a finite state model. With this technique, no reputation information is propagated in the network and malicious nodes cannot cause false penalty to benign hosts. Misbehaving node detection is performed on-demand; and malicious node punishment and avoidance are accomplished by only maintaining reputation information within neighboring nodes. This scheme, however, requires that each node equip with a tamper-proof hardware. In the second technique, no such restriction applies. Participating nodes classify their one-hop neighbors through direct observation and misbehaving nodes are penalized within their localities. Data packets are dynamically rerouted to circumvent selfish nodes. In both schemes, overall network performance is greatly enhanced. Our approach significantly simplifies the collaboration enforcement process, incurs low overhead, and is robust against various malicious behaviors. Simulation results based on different system configurations indicate that the proposed technique can significantly improve network performance with very low communication cost.

To my parents, for their love, encouragements and support

ACKNOWLEDGMENTS

I would like to express my deepest regards and thanks to my academic advisor, Dr. Kien A. Hua for his constant encouragement, inspiration, and high standard. Dr. Hua is a very good mentor as well as a very good friend. As an advisor, Dr. Hua's passion and high standard for academic establish himself a role model for all the students of the Data Systems Group. It has become the DSG culture to always seek originality and work very hard to achieve outstanding research goals. As a friend, I always enjoy our discussions, not only within the domain of computer science, but also on almost every aspect of life. His encouragement and inspiration help me achieve the best of myself.

Secondly, I would like to thank many friends during my study at University of Central Florida. They are Tao Tao, Roy Villafane, Mounir Tantaoui, Wallapak Tavanapong, Ying Cai, Jung Hwan Oh, Tai Do, Duc A. Tran, Fei Xie, Han Yu, Zhiguang Xu, and many more. Tao Tao is my best friend at UCF. He always gives me a hand when I need help. Roy Villafane was the leader of the Oracle project. We spent a lot of time together working on the project and I certainly learned a lot from him. Together we published several papers and I greatly appreciate his excellent work. Mounir Tantaoui is always ready to help others. He proofread many of my papers. I benefit a lot from his comments and inspirations. In addition, we had many interesting discussions on academics and religion. My best regards to Dr. Tavanapong and Dr. Cai. I always remember their sincere encouragement and help. Overall, I have spent the happiest and fruitful moments with them at University of Central Florida.

Finally, I would like to share my greatest happiness with my parents and grandparents. Although they are faraway in terms of physical distance, their love and help are always close by. I cannot emphasize more how much I owe to them for their love and support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS/ABBREVIATIONS

AES	Advanced Encryption Standard
ALOHA	Areal Location Of Hazardous Atmospheres
AODV	Ah hoc On-demand Distance Vector Routing
AUC	Authentication Center
BS	Base Station
BSC	Base Station Controller
CBR	Constant Bit Rate
CDMA	Code Division Multiple Access
CSMA/CA	Carrier Sense Multiple Access/Collision Avoidance
CSMA/CD	Carrier Sense Multiple Access/Collision Detection
CTS	Clear To Send
CW	Contention Window
DCF	Distributed Coordination Function
DIFS	DCF Inter-Frame Space
DSDV	Destination Sequenced Distance Vector
DSP	Detecting STAM Packet
DSR	Dynamic Source Routing
DSSS	Direct Sequence Spread Spectrum
EDGE	Enhanced Data GSM Environment
FDMA	Frequency Division Multiple Access
FHMA	Frequency Hopped Multiple Access

FHSS	Frequency Hop Spread Spectrum
FTP	File Transmission Protocol
GCK	Group Communication Key
GPRS	General Packet Radio Service
GSM	Global System for Mobile
HLR	Home Location Register
HSDPA	High Speed Downlink Packet Access
HSUPA	High Speed Uplink Packet Access
HTML	Hypertext Markup Language
HTTP	Hyper Text Transfer Protocol
IP	Internet Protocol
MANET	Mobile Ad Hoc Network
MAC	Media Access Control
MAC	Message Authentication Code
MC	Misbehavior Counter
MPQ	Module Packet Queue
MSC	Mobile Switch Center
NAV	Network Allocation Vectors
NCUP	Neighbor Classification Update Protocol
OFDM	Orthogonal Frequency-Division Multiplexing
PCF	Point Coordination Function
PDA	Personal Data Assistance
PHY	Physical Layer

PSP	Penalize STAM Packet
PSTN	Public Switched Telephone Network
QoS	Quality of Service
RERR	Route Error
RIP	Routing Information Protocol
RREQ	Route Request
RREP	Route Reply
RRIDR	Route Redirect
RTS	Request To Send
STAM	Finite State Model approach
SIM	Subscriber Identification Module
TCP	Transport Control Protocol
TDMA	Time Division Multiple Access
TD-SCDMA	Time Division Synchronous Code Division Multiple Access
UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunication System
VLR	Visitor Location Register
WAP	Wireless Application Protocol
WML	Wireless Markup Language
WiFi	Wireless Fidelity
WiMax	Worldwide Interoperability for Microwave Access.

WLAN	Wireless Local Area Network
XML	Extensible Markup Language

CHAPTER ONE: INTRODUCTION

This dissertation proposes techniques to enforce collaboration in Mobile Ad Hoc Networks (MANETs). In this chapter, we introduce various wireless technologies and the collaboration enforcement problem. We give high level overview of our techniques and state the contributions of our research. The organization of the dissertation is also outlined.

1.1. Overview of Wireless Networks

In traditional hard-wired networks, computers are inter-connected through hard-wires such as phone lines, cables, and fiber-optical cables. While these networks are usually featured with high reliability and high data transmission rates, they are not easy to configure and users generally lack the capability of moving beyond the physical locations of the network. Nowadays we have seen tremendous growth in various wireless networks. Wireless networks offer the advantage of pervasive connection to the internet and have caused a fundamental revolution to the way of living and doing business. In wireless networks, data is exchanged through electro-magnetic media over a distance through the free-space environment. Users are able to mobile within the duration of data communication. Such flexibility greatly facilitates many important civil and military applications. In this chapter, we will give a review of the contemporary wireless communication networks and techniques. More focus is given to the introduction of Mobile Ad Hoc Networks (MANETs).

1.1.1. Cellular Networks

Figure 1 illustrates a typical cellular network. In a cellular network, large geographical areas are divided into cells. Subscribers within a cell are served by a base station, which transmits and receives radio signals through a fixed transceiver. A certain number of cells are grouped into clusters. All base stations within a cluster are connected to a Mobile Switch Center (MSC) through land lines. Each MSC of a cluster is then connected to the MSC of other clusters and the main switching center of Public Switched Telephone Network (PSTN). The MSC stores information about the subscribers located within the cluster and is responsible for directing calls to them as well as delivering voice data to other MSCs. Equipped with mobile phones or personal data assistance (PDA), subscribers are free to mobile between cells and can make phone calls to other users.

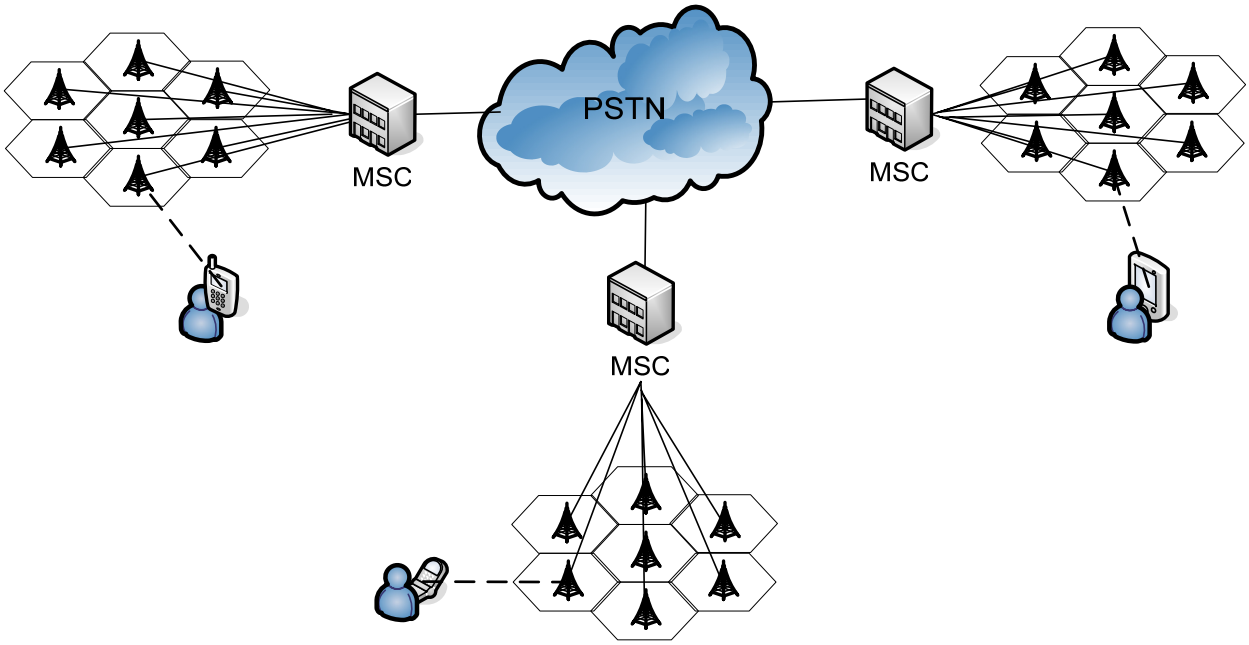


Figure 1. Cellular Networks

An important task of a base station of a cellular network is to allow users share radio bandwidth to increase system capacity. Three multiplexing techniques are employed to achieve this. In Frequency Division Multiple Access (FDMA) [70], the frequency band allocated to a network is divided into sub-bands or channels. Each frequency channel can carry a voice conversation. Each subscriber is assigned a channel for the duration of a call. As a result, multiple users can make simultaneous calls without interfering each other. Alternative multiple access techniques are Code Division Multiple Access (CDMA) [70] and Time Division Multiple Access (TDMA) [70]. In CDMA, the base band signal is multiplied by a very large bandwidth signal called the *spreading signal*. The transmitted signal is then coupled with a pseudorandom codeword unique to each user. The resultant signals of different users are orthogonal so that only the target subscriber terminals can successfully decode. This way, all the terminals in

CDMA can transmit simultaneously and (theoretically) do not interfere with others. TDMA divides each cellular channel into time slots in order to allow users to share the channel. However, in practice TDMA and CDMA are always found in tandem with FDMA to give multiple channels within the coverage area of a single cell.

Modern cellular networks can be chronically classified into 1G (1st generation), 2G, 2.5G, 3G, and 4G networks. First generation cellular systems ([70]) are analog systems, where analog speech signals are transmitted between subscriber terminals to base stations and then digitized to be transmitted over PSTN. The MSC handles all user mobility issues (e.g. handoff) as well as providing network management functions. The 1G systems also support roaming between different providers.

The 2G systems digitize speech signals between subscriber terminals and base stations. 2G systems use dedicated control channels to achieve better channel utilization. A representative example is the Global System for Mobile (GSM) [70]. GSM was first introduced in Europe and is now widely deployed in almost all over the world. GSM employs a TDMA-FHMA (Frequency Hopped Multiple Access) to multiplex subscriber terminals of a cell. GSM's channel data rate is around 14.4 kbps. Another important feature of GSM is that user related information is encrypted and stored on a Subscriber Identification Module (SIM). The SIM approach offers much better security. In addition, SIM cards can be interchangeably plugged into any GSM enabled handset, allowing the subscriber to mobile throughout any GSM system over the world. Furthermore, GSM introduces the base station controllers (BSC) between base stations and MSC to offload the works of MSC. Finally, GSM utilizes the Home Location Register (HLR), the Visitor Location Register (VLR), and the Authentication Center (AUC) to handle roaming. The

U.S. adopts cdmaOne [70] as its 2G system. cdmaOne is implemented based on the IS-95 CDMA standard [27].

2.5G networks in an extension of 2G networks. Features such as packet-switched services and higher data rates are implemented. Representative techniques include Enhanced Data GSM Environment (EDGE) [70] and General Packet Radio Service (GPRS) [70]. Many important services (e.g. email, multimedia message, games) are provided by 2.5G techniques. GPRS supports a data rate of 70-80Kbps whereas EDGE can achieve a data rate of 384Kbps. A family of protocols is defined to support packet switched services for wireless networks. The WAP (Wireless Application protocol) [59] protocol is the leading standard for information services on wireless terminals like digital mobile phones. The WAP standard is based on Internet standards (HTML, XML and TCP/IP). It consists of a WML (Wireless Marked Language) specification, a WMLScript specification, and a Wireless Telephony Application Interface (WTAI) specification. WAP is published by the WAP Forum, founded in 1997 by vendors such as Ericsson, Motorola, and Nokia.

Nowadays, 3G systems are increasingly being deployed to offer higher data rates and packet-switched networks. 3G consists of many parallel standards. Universal Mobile Telecommunication System (UMTS) ([70]) is the 3G successor of GSM networks. It is defined by Europe countries and Japan. UMTS supports a peak data transfer rate of 1920 kbps, much higher than GSM. New techniques such as High Speed Downlink Packet Access (HSDPA) have been proposed to lift the downlink transfer rate to 14.4Mbps. High-Spead Uplink Packet Access (HSUPA) is also underway to enhance uplink data transfer rate. CDMA2000 is a 3G standard designed for 2G CDMA standard IS-95. It offers data rates from 144 kbps to 3Mbps. TD-SCDMA (Time Division Synchronous Code Division Multiple Access) is a 3G standard

developed by China. TD-SCDMA utilizes a dynamic timeslot allocation mechanism to optimize bandwidth usage. Overall, 3G greatly improves 2G techniques and can provide various services that are not possible in 2G systems.

4G [70] is the successor of 3G. It is also dubbed "Beyond 3G" (B3G). 4G will be a packet-switched only network. The prospective data transfer rate of 4G can reach up to 100Mbps and higher. 4G networks are pervasive, where subscribers can seamlessly roam between multiple wireless access networks.

1.1.2. WLAN and WiMax

Besides cellular networks and MANETs, other new communication paradigms keep emerging to support higher mobility and ubiquitous connections. WLAN and WiMax are among the most popular of these techniques.

1.1.2.1. WLAN

A wireless local area network (WLAN) is a local area network that employs radio signals as the carrier. In many cases, wireless access points are deployed to provide stable and high bandwidth services for a certain area. Each access point can service multiple user devices. Nowadays, Wi-Fi (Wireless Fidelity), a set of product compatible standards for WLANs based on the IEEE 802.11 is widely deployed. Some even argue that WiFi might render a significant threat to 3G networks due to its higher data rate and the ever-dropping cost of wireless access points. Whereas others [42] [79] point out that WiFi and 3G are complementary and users can

take advantage of both techniques in different scenarios. In fact, dual-mode devices have been manufactured by many vendors. Their market penetration remains unclear.

1.1.2.2. WiMax

WiMax stands for Worldwide Interoperability for Microwave Access. WiMax is based on the IEEE 802.16 standard, a broadband wireless access standard for systems in the frequency ranges 10-66 GHz and sub 11 GHz. The standard covers both the Media Access Control (MAC) and the physical (PHY) layers.

A number of PHY considerations were taken into account for the target environment. 802.16 works at high frequencies and communication and in most cases require line-of-sight communication. This allows for channels with bandwidth greater than 10 MHz. As a result, 802.16 can provide very high capacity for both uplinks and downlinks.

The MAC layer of 802.16 is designed to offer very high bit rates (up to 268Mbps each way). The MAC frame structure allows terminals to be dynamically assigned uplink and downlink burst profiles according to their link conditions. Moreover, 802.16 MAC frames are of variable length and can be transmitted in a bundle to save physical layer overhead. In addition, payload header suppression exploits the redundant portions of frame headers and can further enhance bandwidth efficiency. Finally, the MAC of 802.16 employs a very flexible bandwidth utilization scheme to eliminate the overhead and delay of acknowledgements, while simultaneously offering better QoS (Quality of Service).

A recent addition to the WiMax standard has added full mesh networking capability. Mesh networks [59] have higher availability due to its link-redundant topology. An example is the community mesh networks. In this scenario, neighbors within a community connect their home networks together. Many advantages come with this technique. For example, when

enough neighbors cooperate and forward each others packets, they do not need to individually install an Internet gateway but instead can share faster, cost-effective Internet access via gateways that are distributed in their neighborhood. Another advantage is that neighbors can cooperatively deploy backup technology and never have to worry about losing information due to a catastrophic disk failure. A third advantage is that this technology allows data created locally and destined to local users to be delivered and consumed locally without having to go through a service provider and the Internet.

1.1.3. Mobile Ad Hoc Networks

A *Mobile Ad hoc Network* (MANET) is a self-organized, highly distributed, and easy-to-configure network formed by devices (often referred to as nodes) equipped with wireless network interface cards. Figure 2 illustrates a typical civilian MANET. We make several important observations from Figure 2. First, nodes that form a MANET are heterogeneous. They can be Personal Data Assistance (PDA), cell phones, laptops, and even digital cameras and cars. Many of the nodes rely on limited battery power and are heterogeneous in terms of size and CPU power. Another important observation is that MANETs do not require dedicated infrastructure. Nodes collaboratively relay data packets for each other. As a result, MANETs require minimal configuration and can be quickly deployed. For example, in Figure 2, node *A* is sending some documents to the printer (node *D*) through intermediate nodes *B* and *C*. Next, although some nodes may be fixed in location, most of the nodes in MANETs can mobile and thus the network's topology may change rapidly and unpredictably. Finally, MANETs may operate in a standalone fashion, or may be connected to the larger Internet through wireless

access points. For example, in Figure 2, node *E* is accessing the Internet through node *F* and the access point.

MANETs open a whole new avenue for various military, emergency, airport, and conferences applications. In this section, we first briefly introduce the history of MANETs. We then discuss the MAC and Network layer of MANETs. Finally, we explain the collaboration enforcement challenge for MANETs.

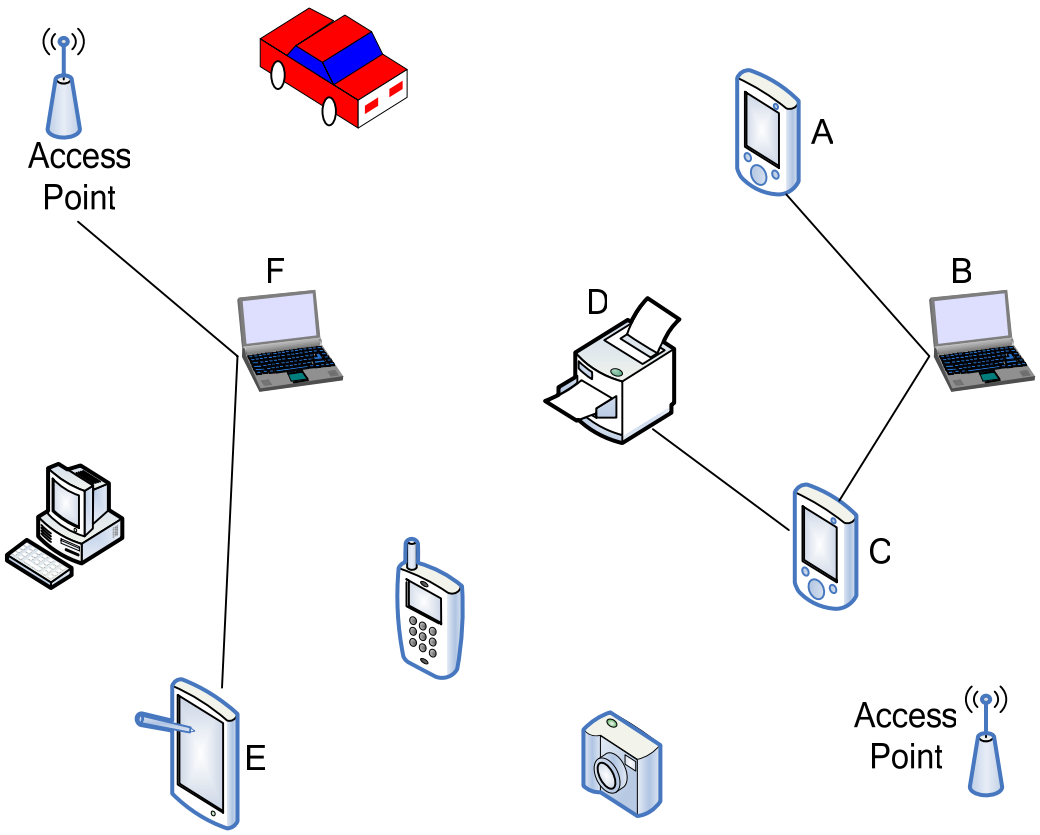


Figure 2. A Civilian Mobile Ad Hoc Network

1.1.3.1. History of MANET

It is well acknowledged that ad hoc networks were pioneered by two independent research projects. In 1968, Norman Abramson of the University of Hawaii pioneered a project named ALOHA (Areal Location Of Hazardous Atmospheres) [80]. The purpose is to develop a network to interconnect various educational facilities. A significant limitation of ALOHA is that it is based on a single-hop protocol. Hence, the source and destination point of communication have to locate within the radio transmission range.

Inspired by ALOHA, in 1973, DARPA started a Packet Radio Network Project, which used broadcast radios for relaying data over multi hop mobile network. The systems were designed and built by BBN Technologies and SRI International. The Packet Radio Network attacked many issues such as media access, routing, mobility, etc. These techniques actually pioneered the original Internet Protocol suite. Details of these issues can be found in [83]. The problems of Packet Radio Networks were subsequently addressed in the Survivable Radio Networks during the 80's.

1.1.3.2. The 802.11 Protocol

The 802.11 [101] family of protocols cover both the physical layer and the MAC layer and form the basis for MANETs. The original version of the standard IEEE 802.11 was released in 1997. 802.11 offers raw data rates of 1 and 2 Mbit/s. At the physical layer 802.11 supports infrared (IR), Frequent Hopping Spread Spectrum (FHSS) and Direct Sequence Spread Spectrum (DSSS). Both FHSS and DSSS work at the 2.4GHz band. The MAC layer of 802.11 defines two access methods: the Distributed Coordination Function (DCF) and the Point Coordination

Function (PCF) although PCF is not widely implemented. Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) was designated as the media access method for DCF. We briefly introduce CSMA/CA as follows.

CSMA is very popular due to the wide deployment of Ethernets. In a CSMA protocol, each node senses the media before it transmits a data frame. If the media is free (i.e., no other node is transmitting simultaneously), the node starts transmission immediately. Otherwise, the transmission is delayed following a random exponential back off mechanism until the media is observed to be free again. Such operations are referred to as CSMA/CD, where CD stands for collision detection. In 802.11, when a node has data ready for transmit, it too senses the media. If the media appears to be clear, the node still has to wait for a period dictated by DCF Inter-Frame Space (DIFS) before any data transmission can take place. If at any time a collision occurs, each colliding node backs off for a time randomly selected within a Contention Window (CW). CW will be exponentially increased if the contention persists. In 802.11, when a node successfully receives a data frame, it acknowledges the sender. If the sender does not receive an ACK from the receiver, it will keep retransmitting the frame until either an ACK is received or a maximum retransmit threshold is hit. Figure 3 illustrates the structure of the 802.11 frame.

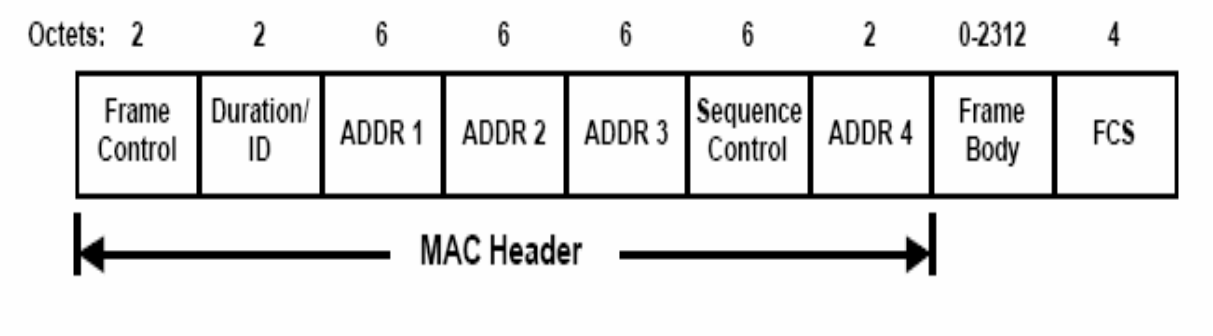


Figure 3. 802.11 Frame

Another problem addressed by 802.11 is the hidden terminal problem. We observe that collision detection works well for wired Ethernets since each node can sense other nodes within the same network. Unfortunately, wireless networks do not have such luxury. Figure 4(a) depicts the hidden terminal problem. In this scenario, both node *A* and node *C* are neighboring nodes (i.e. within radio transmission range) of node *B*. However, node *A* and node *C* are "hidden" (i.e. not within the sense range) to each other. Suppose node *A* and node *C* both want to send data to node *B*. Both nodes attempt to sense the media first. Since they are hidden to each other, both nodes assume the media is free and start transmission. Consequently, the data frames will collide at node *B* and will be corrupted. In certain situations, the hidden terminal problem can greatly hurt network throughput.

(a) Hidden terminal

(b) Collision Avoidance

Figure 4. CSMA/CA

Figure 4(b) shows the solution to the hidden terminal problem. Consider node *A*. Before transmitting a data frame to node *B*, it first broadcasts a short Request To Send (RTS) packet, which includes the source, destination, and the estimated duration τ of the upcoming transmission and acknowledgement. All the neighboring nodes of *A* will record τ in their Network Allocation Vectors (NAV) and pause. When node *B* receives the RTS, if the media is clear, it broadcasts a Clear To Send (CTS) packet, which also includes τ. Upon receiving the CTS, node *C*, as a neighboring node of *B* will record τ in its NAV and refrain from injecting

frames into the network. Essentially, node *A* and node *B* have reserved the media for a time period of τ and are likely to successfully exchange data and ACK frames.

Legacy 802.11 was rapidly replaced by 802.11b since 1999. 802.11b works at the 2.4GHz band. It employs the direct sequence spread spectrum (DSSS) modulation and reaches a maximum raw data rate of 11 Mbit/s. However, due to the CSMA/CA protocol overhead, in practice the peak 802.11b throughput that an application can achieve is about 5.9Mbps over TCP and 7.1Mbit/s over UDP [80]. 802.11b cards will further degrade to 5.5Mbps, 2Mbps, and 1Mbit/s under reduced signal quality.

The 802.11a amendment to the original standard was ratified in 1999. The 802.11a standard uses the same core protocol as the original standard, operates in 5 GHz band, and uses a 52-subcarrier orthogonal frequency-division multiplexing (OFDM) [70], which can achieve a maximum raw data rate of 54Mbit/s. In practice, the achievable throughput is at the mid-20 Mbit/s. Like 802.11b, the data rate of 802.11a can be reduced to 48, 36, 24, 18, 12, 9 then 6 Mbit/s if necessary. Although 802.11a has a higher data transmission rate, it has some drawbacks. First, 802.11a is not interoperable with 802.11b. Second, with higher carrier frequency (5GHz), 802.11a cannot penetrate as far as 802.11b. This has restricted the application of 802.11a to only line of sight scenarios.

1.1.3.3. Routing in MANETs

A prominent feature of MANETs is that they do not rely on infrastructure. This makes MANETs very easy to deploy. However, such benefit is not achieved without cost. The infrastructure-less property along with the inherent mobility of nodes in MANETs makes routing

a challenging task. In wired networks, as the nodes are largely fixed, routing is realized using the notion of location, which is encoded in the IP address. Unfortunately, MANETs do not have such luxury as all the nodes in the network can potentially move and there is no fixed reference point (like a base station) in the network. This forces most of the routing protocols of MANETs to adopt a mechanism called flooding, where routing messages are propagated throughout the network to reach all the participants. Obviously, flooding is very expensive in terms of the number of messages transmitted and should be avoided whenever possible. Most MANET routing protocols are "reactive" in nature in that nodes initiates routing flooding only when necessary (i.e., they need to communicate to other nodes and are not aware of any route to the target). We introduce some MANETs routing protocols in Chapter 2.

1.1.3.4. Collaboration Enforcement in MANET

The easy-to-deploy feature renders itself a double-sided sword for MANET users. On one hand, it brings great flexibility to deploy MANETs in any place with any equipment. On the other hand, the lack of infrastructure and organizational environment offer extra opportunities to attackers. In practice, there can be both selfish and malicious nodes in a mobile ad hoc network. The proper operation of MANETs relies heavily on the collaboration of all participating nodes. Essentially, each node within a MANET is obligated to forward data for others. On the other hand, in many practical scenarios, nodes are restricted in power supply and are thus very sensitive to energy-swallowing operations such as packet forwarding. Obviously, the above two factors form a fundamental conflict, which motivates various selfish behaviors. Selfish nodes are thus most concerned about their energy consumption and intentionally drop packets to save

power. It is acknowledged by many works ([5][6][8][9][10][36][39]) that selfish behaviors (mainly deliberately discarding packets of other nodes) can significantly degrade MANET performance. In fact, lack of motivation for participating nodes to collaborate is very likely to be the major obstacle to the adoption of mobile ad hoc networks. The purpose of malicious nodes, on the other hand, is to attack the network using various intrusive techniques. In general, nodes in an ad hoc network can exhibit Byzantine behaviors. That is, they can drop, modify, or misroute data packets. As a result, the availability and robustness of the network are severely compromised.

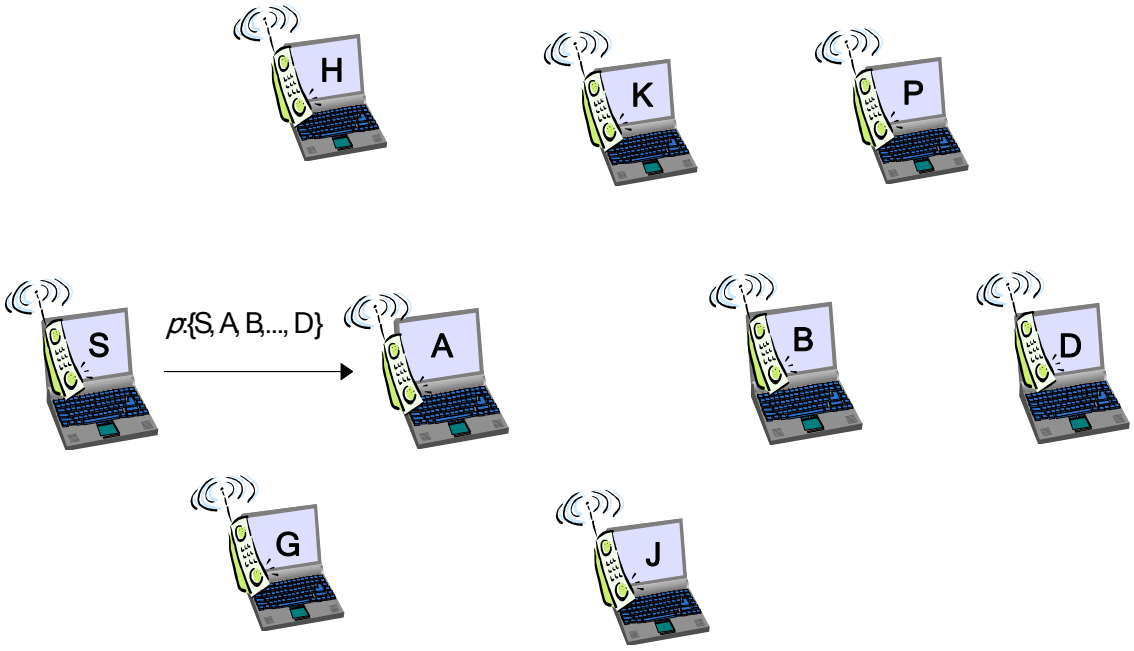


Figure 5. Selfish Behavior in MANETs

In our research, we mainly focus on selfish nodes as they can greatly reduce the availability of MANETs. Selfish behavior can take place in multiple layers. We are most

concerned about the network layer as it is more vulnerable than the MAC layer and routing failure can cause more negative impact to MANETs. At the network layer, selfish behaviors can happen in both routing discovery phase and data transmission phase. During routing discovery, selfish nodes can cause failure in propagating routing messages and result in suboptimal routes. Figure 5 illustrates the selfish behavior during data transmission. Suppose node *S* is transmitting data packets to node *D* through the route {*S*, *A*, *B*, *D*}. We can see that any selfish node on the route can render the whole route unusable.

It is observed that selfish nodes can gain advantage over other benign nodes as they do not forward data whereas other nodes still relay their data. In other words, nodes in MANETs do not have the motivation to collaborate. This is a fundamental problem of MANETs. The focus of most collaboration enforcement solutions is to create incentives to motivate collaboration. Details will be discussed in Chapter 2.

1.2. Overview of The Proposed Approach

Many techniques have been proposed to enforce collaboration in MANETs. Most of these techniques employ reputation mechanisms for nodes to avoid and penalize malicious participants. Reputation information is propagated among network participants and updated based on complicated trust relationships to thwart false accusation of benign nodes. The aforementioned strategy suffers from low scalability, high communication overhead, and is likely to be exploited by adversaries.

In this dissertation, we propose a novel approach to address these drawbacks. In our approach, no reputation mechanism is propagated in the network. Nodes classify their one-hop neighbors through direct observation and misbehaving nodes are penalized within their localities. This greatly reduces the cost of the collaboration enforcement algorithm, and eliminates the possibility of any attacks targeted at the reputation propagation mechanism. We develop two schemes for the approach. In the first scheme,

- A tamper proof module is equipped by each participating node in MANETs.
- A finite state model detection and penalty mechanism is employed. With this technique, misbehaving nodes detection is performed on-demand; and malicious node punishment and avoidance are accomplished by only maintaining reputation information within neighboring nodes.

In the second scheme,

- No tamper proof module is required.

- A dynamic rerouting mechanism is proposed to help nodes avoid misbehaving nodes. The dynamic rerouting mechanism is also fortified. As a result, overall network performance is greatly enhanced.

Our approach significantly simplifies the collaboration enforcement process, incurs low overhead, and is robust against various malicious behaviors. Simulation results based on different system configurations indicate that the proposed techniques can significantly improve network performance.

1.3. Dissertation Organization

The remainder of this dissertation is organized as follows. We discuss related works in Chapter 2. In Chapter 3, we introduce the first scheme and the experimental results. In Chapter 4, we present the second scheme. Finally, we conclude in Chapter 5.

CHAPTER TWO: LITERATURE OVERVIEW

In this chapter, we first review some of the routing protocols designed for MANETs. We then introduce the literatures of security in MANETs. Finally, we present some important research on collaboration enforcement in MANETs.

2.1. Routing in MANETs

In MANETs, there is no dedicated infrastructure deployed to relay data for participating nodes. When two nodes that are not within the wireless transmission range of each other want to exchange data, they have to identify a route consisting of one or more intermediate nodes before any data transmission can take place. Numerous routing protocols have been proposed for MANETs. These protocols can be categorized into three families: proactive [15] [64] [66] [73] [38] [20] [57], reactive [23] [82] [67] [23] [87] [40] [46] [13] [51] [65] [75], and hybrid [26]. Furthermore, some routing techniques specifically take energy consumption [48] [89] or geographical location into consideration [44] [11].

In proactive routing protocols, nodes actively discover and maintain routes to other nodes, just like their counterparts in hardwired networks. However, these protocols generally incur high energy cost and are not very suitable for MANETs. Reactive routing protocols were proposed to address the problem. In reactive routing protocols, a node does not bother to seek for a route until it actually intends to transmit data to other nodes. In this dissertation, we briefly introduce a proactive routing protocol, the Destination-Sequenced Distance Vector (DSDV) protocol ([66]) and two reactive routing protocols, the Dynamic Source Routing (DSR) protocol

([40]) and the Ad-hoc On-demand Distance Vector (AODV) protocol ([67]). An overview of various routing protocols can be found in [50].

2.1.1. DSDV

A representative of proactive routing protocols is the DSDV protocol. DSDV is a derivation of the distance vector routing protocol [80] in wired networks. In general, the purpose of a distance vector routing protocol is to calculate shortest paths between each pair of nodes in the network through a distributed implementation of the classical Bellman-Ford algorithm [80]. Distance vector protocols are easy to implement and are efficient in terms of memory and CPU processing capacity. A popular example of a distance vector routing protocol in the wired network world is the Routing Information Protocol (RIP) [104] [105].

In the basic version of distance vector routing, each router employs a routing table to maintain all the possible destinations within the network. Each entry of the routing table contains the IP address of a destination node, the shortest known distance (usually in number of hops) to that destination, and the address of the neighboring router that is the first hop on this shortest route to that destination; the distance to the destination is known as the *metric* in that table entry. To maintain the routing tables, each node periodically transmits a routing update to each of its neighboring routers. The routing update includes the metric to each destination node from a particular node. Each node uses the routing updates advertised by its neighbors to recalculate the optimal paths to the destinations. A common optimization to this basic procedure to spread changed routing information through the network more quickly is the use of *triggered updates*, in which a node transmits a new update regarding some destination as soon as the metric in its table entry for that destination changes, instead of waiting for its next scheduled

periodic update. A problem for the basic distance vector routing protocols is the existence of loops for some destination. A loop-free route might be achieved through many rounds of routing table updates (referred to as the "counting to infinity"). The counting to infinity problem is caused by the fact that the routing updates do not include the exact path to the destination node. As a result, neighboring nodes have no way to resolve route loops. Solutions can be found in [104] [105] [17] [41].

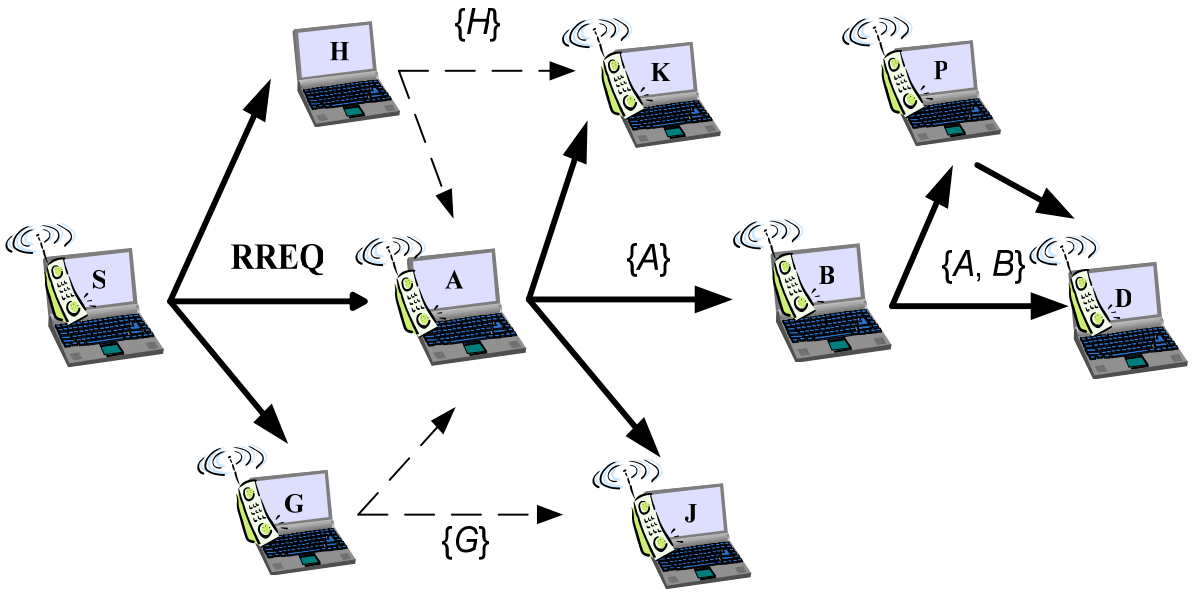
When routing a packet to some destination, the node transmits the packet to the indicated neighboring router, and each router in turn uses its own routing table to forward the packet along its next hop toward the destination.

When applying distance vector routing In MANETs, the route loop problem can be more common due to the motion of the nodes and the changes in wireless channel conditions. To address the problem, DSDV includes a *sequence number* in each routing table entry. The sequence number helps filter out routing loops caused by out-of-order updates, which is quite common in MANETs as the routing information may spread along multiple paths through the network. In DSDV, each node attaches an *even* sequence number to each routing update it originates, and each entry in a node's routing table is marked with the most recent sequence number. When a node detects a broken link to a neighbor, it starts a new routing update with an "infinite" metric and the next *odd* sequence number after the even sequence number in its corresponding routing table entry. When a node receives a routing update, for each destination in the update, the node accepts either a "fresher" advertised route (i.e. the sequence number associated with the route is greater than in the corresponding entry currently in the node's routing table), or if the sequence numbers are equal but the new metric is lower than in the

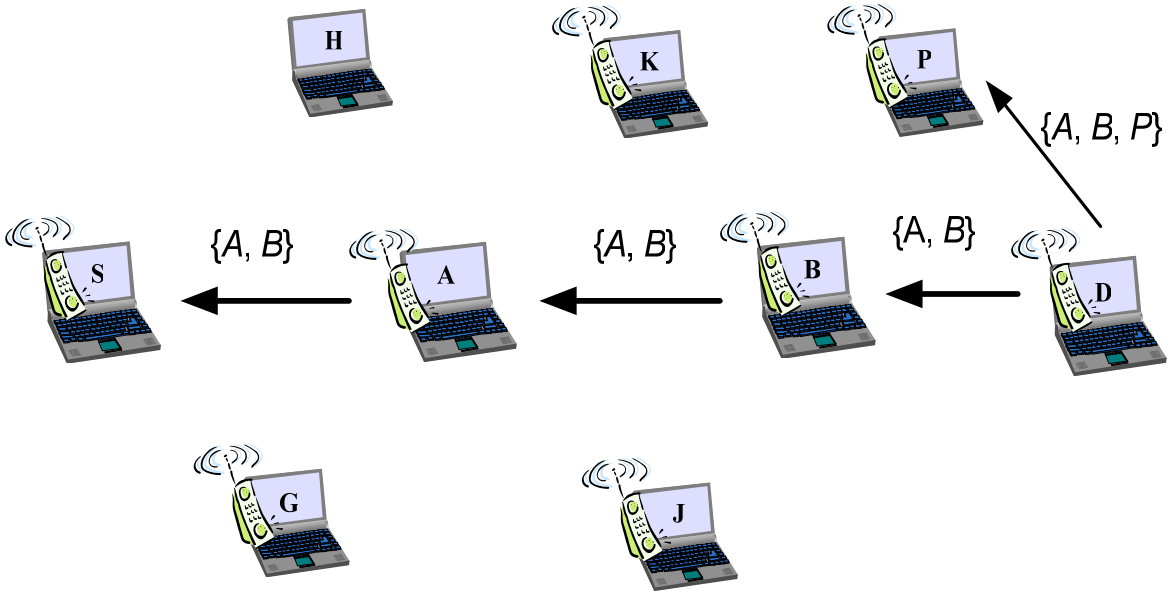
node's current table entry for that destination. If the sequence number in the update is less than the current sequence number in the table entry, the new update for that destination is ignored.

2.1.2. DSR

DSR is a reactive routing protocol. It is based on the concept of source routing. To reduce the number of route discoveries, each node maintains a route cache that stores all the routes it knows. DSR consists of two phases: *route discovery* and *route maintenance*. When a node needs to transmit data packets to another node, it first checks its route cache. If a route exists in the route cache, the node will send the data packets using the route. On the other hand, if it cannot find a route to the destination in the route cache, it invokes the route discovery process. In the route discovery phase, the source node broadcasts a *Route Request* (RREQ) packet to all its neighbors. The RREQ packet includes the addresses of the source and destination nodes, and a unique identification number. Each neighboring node appends its address to the packet and broadcasts it if the node is not the intended destination and it has not seen the RREQ packet before (by inspecting the identification number). The latter condition helps to limit the number of outgoing RREQ packets. RREQ packets are flooded in the network until they reach a node that is aware of a route to the destination.



(a) Route Request



(b) Route Reply

Figure 6. Route Discovery

A *Route Reply* (RREP) packet will be generated either by the destination node or an intermediate node that is aware of a route to the destination. In the former case, the RREQ includes the complete path and the destination node simply copies the whole path to the RREP packet. In the latter case, the intermediate node will have to append the route to the destination to the route stored in the RREQ packet. In either case, the RREP packet will include the complete route. The RREP packet needs to be delivered back to the source node. In the case that symmetric links are supported, the RREP packet will be transmitted along the recorded route in reverse order. Otherwise, it will be included in a RREQ packet and broadcasted back to the source node. Generally, the source node may receive multiple routes, from which it selects the best one (by default, the shortest route) for data transmission.

Besides discovering routes through route discovery, nodes also extract paths from snooped RREP and data packets.

Figure 6 illustrates the route discovery of DSR. Suppose node *S* wants to discover a route to node *D*. Figure 6(a) illustrates the route discovery phase. Node *S* first broadcasts a RREQ packet, which will be received by its neighboring nodes, *A*, *G*, and *H*. *A*, *G*, and *H* will all attach their IP address and propagate the RREQ. Suppose *A* capture the media first. It will broadcast a RREQ to its neighbors, *B*, *J*, and *K*, and so on. Eventually the RREQ will reach *D*, the destination node. And the route is embedded in the RREQ packet. We note that *A* will discard the RREQ sent by node *G* and *H*, as provisioned by the DSR protocol.

Every time *D* receives a RREQ packet, it constructs a RREP packet and attaches the route from the received RREQ packet to the RREP packet. Figure 6(b) illustrates the route reply process. Suppose D first receives a RREQ with an embedded route $\{A, B\}$. It will instantiate a RREP packet and transmits it to node *B*. Node B inspects the RREP packet and realizes that it

should forward the RREP packet to node *A*, and so forth. Eventually node *S* will receive a RREP packet with route {*A*, *B*}. *S* will store the route in its route cache. Similarly, in Figure 6(b), S will receive another RREP packet with route {*A*, *B*, *P*}. In DSR, *S* will always pick the shortest path to deliver the data packets. When node *S* transmits data, it attaches the selected route to the destination node ({*A*, *B*, *P*}) in each data packet. Intermediate nodes relay packets according to the embedded source routes.

In the route maintenance phase, when a node identifies a failed link, it sends a Route Error (RERR) packet to the source of the route, which removes all the routes containing the broken link from its cache. Furthermore, the node tries to salvage the packet by looking for a route to the destination in its own route cache.

2.1.3. AODV

In AODV, each node maintains a route cache. When a source node has data to be sent to some destination node and does not already have a valid route to that destination, it initiates a *path discovery* process as follows. The source node broadcasts a RREQ message to its neighboring nodes. The RREQ is enclosed with a monotonically increasing *broadcast ID* and the latest known sequence number to the destination. The broadcast ID and the IP address uniquely identifies a RREQ packet. Intermediate nodes can only respond to the RREQ if they have a route to the destination and the sequence number corresponding to the route is greater than or equal to the one attached in the received RREQ.

During the propagation of the RREQ, intermediate nodes record in their route tables the address of the neighbor from which the first copy of the broadcast packet is received, thereby establishing a reverse path to the source node. Any duplicate copy of the same RREQ received later will be discarded subsequently. The RREQ is broadcast until it reaches either the destination node or a node that has a route to the destination with the destination sequence number higher than that enclosed in the request. The destination/intermediate node responds by unicasting a route reply (RREP) packet back to the neighbor from which it first received the RREQ. As the RREP is routed back along the reverse path, nodes along this path set up forward route entries in their route tables which point to the node from which the RREP was received. As a result, a forward route to the destination node is established. Each route entry is timed. Any route that has been idle for a certain amount of time will be purged from the route table.

We note that the RREP is forwarded along the reversed path where RREQ packets have traversed. Thus, AODV can only support symmetric links.

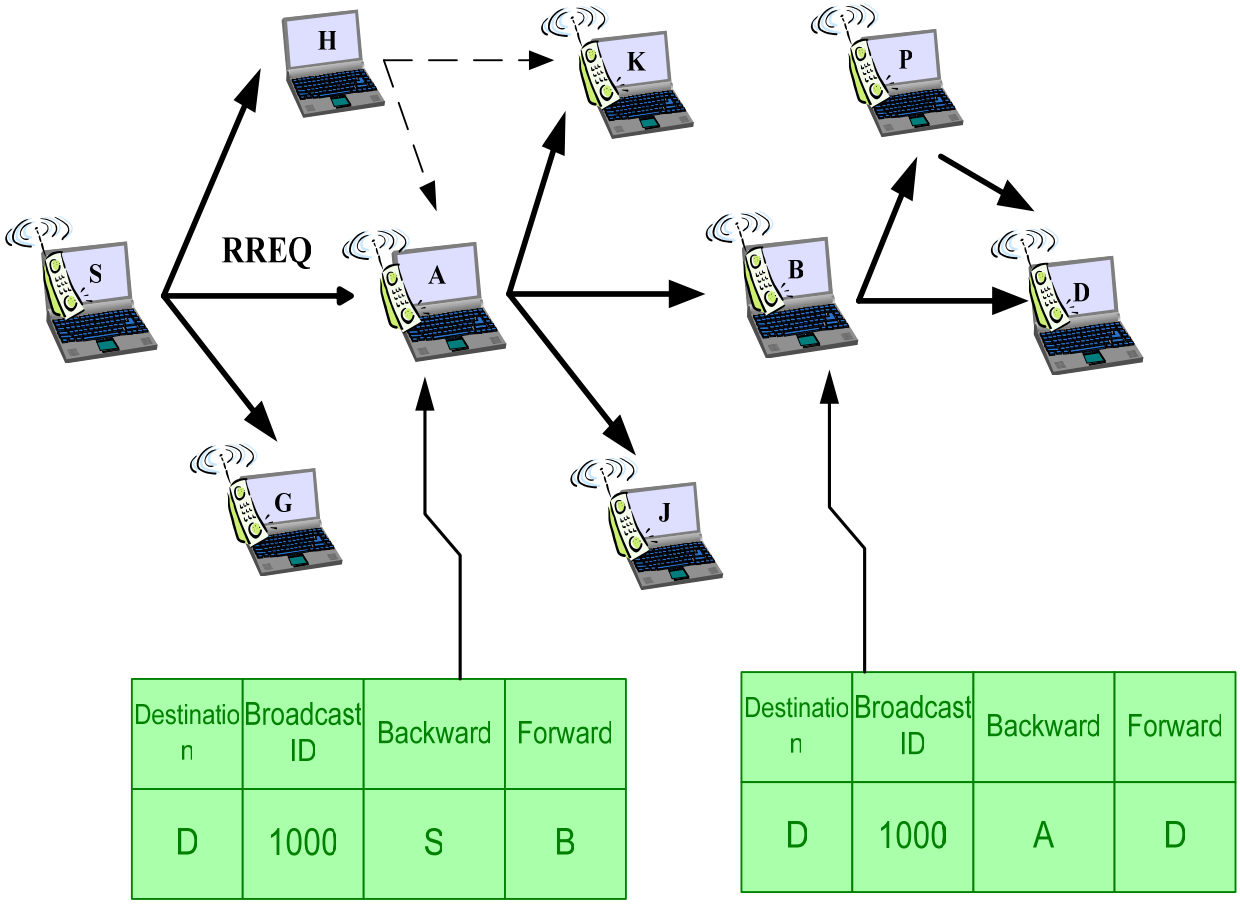


Figure 7. AODV

Figure 7 depicts the route discovery process of AODV. Suppose node *S* is to transmit data to node *D*. Once *S* realizes that it does not have a route to *D* in its route table, it broadcasts a RREQ to its neighbors. We consider node *A*. Suppose node *A* has never received the RREQ before. It records in its route table the destination node, *D*, and the node from which it receives the RREQ packet, *S*. *A* then forwards the RREQ to its neighboring nodes. We note that *A* will ignore all the subsequent RREQ packets it receives (probably from *H* and *G*). The RREQ will be processed by node *B* likewise. Eventually, the RREQ reaches the destination node, *D*, which records the first node that relays the RREQ (*B* in our example). Now, *D* initiates a RREP and transmits it back to *B*. Upon receiving the RREP packet, *B* records the sender (node *D*) of the RREP packet in its route table and relays the RREP to its previous node (*A* as recorded). Similar operations will be performed by node *A*. Finally, the RREP packet is back to *S* and the route to *D* is established.

Route maintenance of AODV is performed as follows. If an intermediate node of a route moves away, its immediate upstream node will realize this and instantiates a link failure notification. The link failure notification message will be propagated back to the source node. During the propagation, all the nodes update their route tables. The source node might need to start a new route discovery if the route to the destination node is still in need.

2.2. Routing Security for MANETs

As we have mentioned in Chapter 1, MANETs are extremely vulnerable to various malicious behaviors. In particular, the network layer is a common target of attacks as it offers abundant opportunities for malicious users to play around.

Many attacks have been identified in numerous literatures [29] [31] [35] [60] [62] [59]. Possible attacks against the routing mechanism of MANETs include:

- Eavesdropping the wireless media. Such attacks can usually help the attackers compromise the privacy and anonymity of participants.
- Injecting fabricated routing information. Using this technique, attackers can create route loops, causing packets to be transmitted in cycles without reaching the destinations. Unnecessary energy and bandwidth will be wasted.
- Creating blackholes. An attacker can advertise itself on a (fabricated) optimal route. Thus, all the network traffics will be attracted through it. It can then drop all the packets, thereby forming a "blackhole" in the network.
- Creating wormholes. A wormhole is created by two colluding malicious nodes. These two nodes might locate in different parts of the MANET. When a node receives a packet, it tunnels the packet to the other node, thus creating a wormhole. When applying this attack to the routing protocol, colluding nodes can effectively create any fabricated route and render the routing mechanism the victim of the attack.

- Rushing attack. This attack can be launched against any protocol that either implements a suppression function for duplicate flooding packets or employs a back-off mechanism for contention resolution. In this attack, an adversary "rushes" a rogue packet to a destination (possibly to an intermediate node on the path or to a destination), making the legitimate packet look like a duplicate and be discarded. The rushing effect is achieved by disregarding the delay required either by the MAC layer or the routing protocol. The attacker can thus successfully disrupt routing in MANETs.
- An attacker can also partition the network by injecting faked routing packets. This will effectively block data transmission.
- An attacker can drop or advertise a longer path, thereby forfeiting the responsibility to participating in any routing efforts for other nodes.

Many techniques have been proposed to fortify the routing protocols for MANETs. In [29] [30], the authors present the design and performance evaluation of a new on-demand secure ad hoc network routing protocol, called Ariadne. Ariadne relies solely on highly efficient symmetric cryptography ([80]) and is thus very efficient. In Ariadne, nodes authenticate routing messages using one of three schemes: shared secrets between each pair of nodes, shared secrets between communicating nodes combined with broadcast authentication, or digital signatures. The papers devote more efforts in employing TESLA [63] [69], an efficient broadcast authentication scheme that requires loose time synchronization to secure DSR.

TESLA avoids the need for synchronization, but at the cost of higher key setup overhead. TESLA computes a message authentication code (MAC) [80] to a message for broadcast authentication. Generally, a MAC is computed by a node using a secret key. Any other node knowing the secret key can verify the authentication of a message. We note that routing packets

are broadcast in the network. One way to secure broadcast communication using MAC is to have multiple receivers maintain the same MAC key for verification. However, this would also allow these receivers to forge packets and impersonate the sender. TESLA differs from traditional asymmetric protocols such as RSA [70] in that TESLA achieves this asymmetry from clock synchronization and delayed key disclosure, rather than from computationally expensive functions. With ARIADNE, malicious nodes cannot remove other nodes in a route. Neither can it add any nodes to the route to cause route loops or make the route invalid. Both the source node and the destination node can verify the integrity of the RREQ and RREP packets. The downside is that the technique requires clock synchronization. Moreover, not allowing intermediate nodes to generate legitimate RREP degrades the route discovery efficiency.

In [33], the authors present a technique called "packet leash" to prevent wormholes. The idea is to restrict the packet's maximum allowed transmission distance by employing either "geographical leases" or "temporal leashes". A geographical leash restricts the propagation distance of a packet. A temporal leash regulates the maximum allowed lifetime of a packet. When a node transmits a packet, it includes either the geographical leash or the temporal leash, and signs the packet. A receiver can verify whether the packet is legitimate by inspecting the embedded leash information and the speed of the light. This approach requires time synchronization among all the participating nodes of a MANET.

In [34], a technique is proposed to combat the rushing attack. The scheme employs a secure neighbor detection mechanism to verify that neighboring nodes of a node is indeed within its proximity. A source node permits one of its neighboring nodes to forward a RREQ by signing and sending a Route Delegation Message. The RREQ will be forwarded if the Route Delegation Message is successfully authenticated. Furthermore, a randomized RREQ

forwarding technique is employed to replace the duplicate RREQ suppression mechanism in conventional routing protocols. All these techniques jointly defeat rushing attacks.

Many other techniques have been proposed to defeat the aforementioned routing misbehaviors. In [31], the authors introduce a technique (referred to as SEAD) to defend distance vector routing protocols form various attacks. In [62], the authors propose a technique (dubbed SLSP) to secure link state routing protocols in MANETs. In SLSP, each node constructs a secure neighborhood within R hops using a secured Neighbor Lookup Protocol (NLP). Each neighboring node is identified by both its IP address and MAC address. Nodes periodically broadcast signed link state updates. Only correctly verified information will be accepted. SLSP does not rely on a central key management server. Nodes periodically broadcast their certified public keys within their neighborhood. SLSP is proved to be resilient to individual Byzantine attacks.

In [60] [61], two schemes were proposed to secure the route discovery and data packet transmission, respectively. The Secure Routing Protocol (SRP) proposed in [60] assumes that the source node and the destination node share a symmetric security key. The source node computes a MAC for the RREQ. Furthermore, intermediate nodes are not allowed to reply from their route caches. All the RREP packets must be generated by the destination node. In generating the RREP, the destination node also attaches the MAC of the RREP. When the source node receives the RREP, it will accept the RREP if successfully authenticated. It is demonstrated that SRP can defeat many routing disruptions. In [94], a technique is proposed to protect the AODV protocol. The scheme employs hash chain and digital signatures to protect the mutable and immutable parts of the routing packets. Intrusion detection techniques are adopted to protect MANETs in [63] [84] [96]. Intrusion detection system can identify the anomalies

during routing and data transmission. A cooperative secure protocol is proposed in [49], where trusted nodes within a domain jointly enhance the security of MANETs. [85] [93] and many other papers present other security aware routing techniques. [46] [12] [36] [77] [99] and many other works propose many techniques to secure the key management process, which is a fundamental requirement for most security approaches.

2.3. Collaboration Enforcement

The works introduced in 2.2 address the generic security issues for route discovery in MANETs. Very few of them are concerned about selfish behaviors occurred during route discovery as well as data transmission. In this dissertation, we particularly focus on the collaboration enforcement problem of MANETs since the flexibility and proper operation of MANETs rely heavily on the collaboration of all participating nodes to relay data for others. As we pointed out in Chapter 1, in many practical scenarios, nodes are restricted in power supply and are thus very sensitive to energy-swallowing operations such as packet forwarding. Obviously, the above two factors form a fundamental conflict, which motivates various selfish behaviors. It is acknowledged by many works ([5] [6] [36] [31] [30] [42] [60] [61] [74] [99] [90] [16]) that selfish behaviors (mainly deliberately discarding packets of other nodes) can significantly degrade MANET performance. In fact, lack of motivation for participating nodes to collaborate is very likely to be the major obstacle to the adoption of MANETs.

In this dissertation, we focus on both selfish and malicious nodes. A selfish node attempts to avoid the responsibility of forwarding data in two ways. First, by not participating in route discovery, a node can greatly reduce the chance of being selected to forward data packets by other nodes. Second, a selfish host can cooperate in route discovery, but subsequently discards data packets to save energy. It is pointed out in [55] that the latter misbehavior has more negative impact on overall network throughput. We also detect and penalize malicious nodes that attempt to disrupt data transmission on the MAC layer.

The current state of the art in enforcing collaboration in mobile ad hoc networks can be categorized into three groups, namely incentive motivation approaches, game theory based approaches, and misbehavior penalty approaches.

We consider incentive motivation techniques first. The authors of [8] [9] [63] proposed to use virtual currency to stimulate incentives for nodes to cooperate with each other. In their techniques, each node maintains a "wallet" of nuggets. A node has to possess enough nuggets to reward other nodes for relaying its packets. The only way a node can gather enough nuggets is to forward packets for other nodes. This technique relies on a tamper proof security module and cryptographic techniques to prevent possible abuse of the protocol. The authors of [100] also propose a virtual currency technique that can stimulate node collaboration and defeat colluding malicious users without employment of the tamper resistant module. This scheme requires that each node reports a signature of each packet it forwards to a *Central Clearance Service*. The practicability and performance of this approach remains unclear.

Another class of schemes ([19] [76] [97] [98]) utilizes game theory [57] to model the cooperation enforcement problem in MANETs. Essentially, the purpose of these techniques is to derive strategies that consist of Nash equilibrium. Under the Nash equilibrium, no player (nodes) can benefit from violating the proposed strategy. In [19], a virtual currency approach based on a mechanism design technique is presented. The goal of a mechanism design technique is to define a game played by independent agents according to the rules set by the mechanism designer such that the desired outcome, called the social optimum, can be achieved. This technique guarantees that nodes cannot gain anything through cheating during application data delivery. In [97] [98], a similar technique is proposed to support multicast in MANETs. In [76], an algorithm based on the Generous TIT-FOR-TAT (GTFT) strategy [2] is proposed. The

authors prove the Nash equilibrium of the strategy. In general, the game theory based approaches assume nodes are rational (i.e. their behaviors are determined by their self interests) and are usually not robust to malicious participants (i.e. nodes willing to sacrifice their own benefits to cause devastating results to MANETs).

Our research, on the other hand, falls in the third category. The main idea is to detect, penalize, and avoid malicious and selfish hosts in MANETs. In [96], the authors use intrusion detection techniques to locate misbehaving nodes. A watchdog and a path rater approach is proposed in [52] to detect and circumvent selfish nodes. The main drawback of this approach is that it does not punish malicious nodes. This problem is addressed in [4] [5] [6] [7]. The approach, called CONFIDANT, introduces a reputation system whereby each node keeps a list of the reputations of others. Malicious and selfish nodes are detected and reputation information is propagated to "friend" nodes, which update their reputation lists based on certain trust relationships. During route discovery, nodes try to avoid routes that contain nodes with bad reputations. Meanwhile, no data forwarding service is provided for low reputation nodes as a punishment. Another reputation-based technique, called CORE, is proposed in [53]. In this scheme, only positive reputations are disseminated. A formal analysis of CORE is given in [56]. A trust evaluation technique is proposed in [81]. In [1], the authors attack the problem of defending application data transmission against Byzantine errors. In their approach, each node maintains a weight list of other nodes. Malicious nodes are located by an on-demand detection process and their weights are increased consequently. A routing protocol is designed to select the least-weight path between two nodes. This approach is also based on per-node reputation lists. In addition, the detection process requires that each intermediate node transmit an acknowledgement packet to the source node. In [28], reputation information is propagated

locally and one-way hash functions are employed to secure reputation propagation. In [89], the authors propose an approach that does not assume any a priori trust relationship between nodes in MANETS. Each node has to obtain a token jointly issued by its neighbors in order to be admitted to the network. In [45], the authors propose to use "self-healing communities" to mitigate selfish nodes. The approach requires modification of underlying routing protocol and overhead to maintain the communities. In [39], a finite-state-model technique is introduced. The technique requires that nodes install tamper-proof modules. Reputation packets are only broadcast locally.

In general, most of existing detection and reaction techniques based on reputation dissemination mechanisms suffer from the following drawbacks:

- Reputation-propagation-based schemes have low scalability. Generally, quite a few reputation packets need to be propagated before "bad citizens" of MANETs can be captured, avoided, and punished. As a result, the cost of cooperation enforcement is quite high.
- Reputation-propagation-based schemes offer incentives to various attacks. Most prominently, malicious users can "poison" the reputation lists by disseminating incorrect reputation information. Such packets can be spoofed with other nodes' addresses to hide the identity of the attacker or to pretend to be a "friend" of the receiver. In [1], digital signatures and message authentication codes [102] are employed to defeat packet spoofing. However, if a host is possessed (or physically captured) by a malicious user, cryptographic information of the particular node can be extracted and reputation poison attacks can still be mounted. In [7], the same authors of the CONFIDANT protocol present a scheme based on the Bayesian inference

model to reduce false accusations. This scheme achieved significant reduction in false accusations for some types of reputation poisoning strategies but failed in some others. However, the scheme still relies on flooding reputation information and does not address the scalability concern.

CHAPTER THREE: THE FINITE STATE MODEL APPROACH

In this chapter, we introduce a technique based on a finite STAte Model. Hereafter, we refer to this approach as the STAM approach.

3.1. Node Configuration and Tamper Proof Module

Before presenting the proposed technique, we first describe the configuration of mobile ad hoc nodes in our scheme.

3.1.1. Node Configuration

The proposed technique is based on nodes with the following configurations. First, nodes are equipped with wireless interface cards that can be switched to promiscuous mode to "hear" data transmission in their proximities. Second, reactive routing protocols are employed in the network layer. According to many studies [50] [40], reactive routing protocols are more suitable for MANETs than proactive protocols. We note currently there is no industry standard for MANET routing protocols. Without loss of generality, we base our discussion on the DSR. Nevertheless, the technique can be incorporated into any standard protocols to protect nodes against uncooperative behaviors. Third, reliable communication protocols such as TCP are employed in the transport layer. In reliable protocols, receivers acknowledge senders to confirm

the successful delivery of data packets. Most popular services such as TELNET, FTP, and HTTP are based on the TCP protocol. Finally, nodes employ a HELLO protocol to discover and establish shared secret keys with their neighbors. The HELLO protocol runs at the network layer and periodically generates heartbeat messages. We denote the heartbeat message transmission interval as τ.

3.1.2. Tamper Proof Module

Tamper proof modules are employed by various applications from credit cards to Subscriber Identity Module of mobile phones to achieve security. It is very likely that mobile ad hoc nodes will adopt the same approach, as pointed out in [5] [8] [9]. Like the technique presented in [8], we also equip each node with a tamper resistant module. All other hardware and software components are susceptible to illicit modifications. Our approach guarantees that as long as the tamper resistant module is not compromised, nodes cannot benefit from uncooperative behaviors.

3.1.2.1. Protected Data

Some mission critical data is stored in the tamper resistant module as follows:

- Unique identifier. Each tamper proof module has a system-wide unique identifier, defined as ID_A, where A represents the node on which the tamper proof module is installed.

- Public key/private keys. The tamper proof module is issued a public key and a corresponding private key. Since public key encryption and decryption are usually an order of magnitude slower than secret key schemes with similar security capability, they are only used rarely (during establishment of shared secret key for two neighboring nodes) in the proposed scheme. Of course, this requires a public key infrastructure (PKI) in MANETs. Several techniques [99] [36] have been proposed to address the problem.
- Shared secret keys. When two nodes *A* and *B* become neighbors, their tamper resistant modules establish a shared secret key using public key cryptography [80].
- Group communication key (GCK). Critical information originated by a node *A* is always protected by a unique group communication key (GCK_A) of *A*. Once *A* has established a shared secret key κ_{AB} with another node *B*, it will encrypt GCK_A using κ_{AB} and sends it to *B* and vice versa. This way, any packet encrypted with GCK of *A* can be interpreted by *all* its neighboring nodes. We note that there are other group key distribution techniques such as [78] [37] [92]. However, this topic is beyond the scope of this dissertation and the naïve approach is adopted.
- Counters. A pair of counters is maintained by a node for each of its neighboring nodes to defeat message replay attacks. One of them is the sending counter and the other is the receiving counter. The counters are initialized during the exchange of Heartbeat messages.
- Current node state. At any time, each node is in one of four states of a finite state model. The current state of the node is stored in the tamper proof module so that a malicious node cannot contaminate.

- Module Packet Queue (MPQ). State transition packets generated by the tamper proof module are stored in a queue. They will be transmitted together with heartbeat messages.
- Neighboring node list. A list of neighboring nodes is maintained for each node by its tamper proof module. A misbehavior counter (MC) is associated with each neighboring node to facilitate misbehaving node detection. In addition, the tamper proof module of a node classifies neighboring nodes into different types to defeat various attempts to circumvent the proposed technique.

Since the tamper proof module maintains information of the finite STAte Model, hereafter, we also refer to it as the STAM module.

3.1.2.2. STAM Module Operations

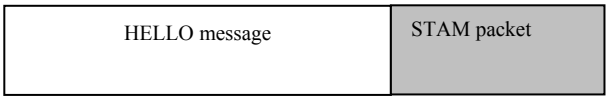
The STAM module provides the following important services.

- It investigates *all* the packets (e.g. routing packets, data packets, heartbeat messages, etc.) exchanged between the network layer and the MAC layer.
- It constructs STAM packets as depicted in Figure 8(b). Each node propagates its current state to its neighboring nodes through the S_1 field of a STAM packet. STAM packets are also sent to initiate state transition of specific neighboring nodes when necessary. The STAM module stores STAM packets in the MPQ. Table 1 lists STAM packets and their definitions.
- Once an outgoing HELLO packet is submitted to the STAM module from the network layer, the STAM module 1) pops up a STAM packet from the MPQ if the MPQ is not empty, or generates a dummy STAM packet (i.e. a STAM packet with its *type* field set to 0) otherwise; 2) copies the sending counter to the *counter* field of the STAM packet; 3) encrypts the STAM packet using the GCK of the sender; and 4) attaches the encrypted packet to HELLO packets. Figure 8(a) depicts the resultant packet. We note that by storing cryptographic information on tamper proof modules, no STAM packet falsification can be performed by malicious users. Moreover, the *counter* field of the STAM packet can be utilized to defeat message replay attacks.
- The STAM module classifies neighboring nodes according to the STAM packets it received. Details are presented in Section 0.

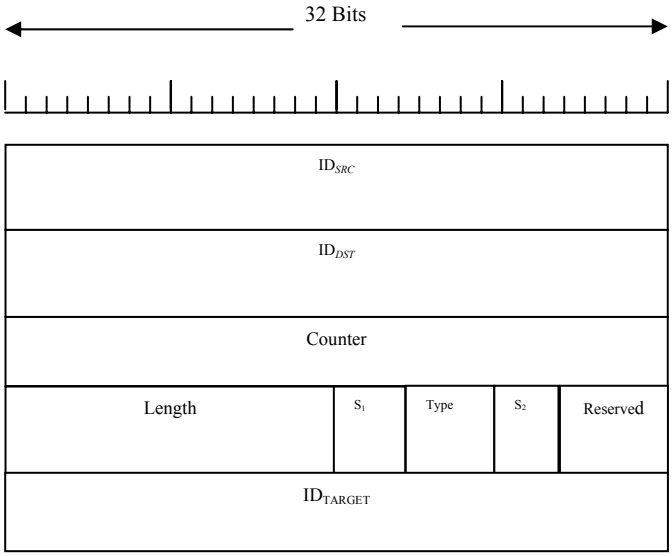
The STAM module performs state transition upon decryption and authentication of STAM packets.

Table 1. STAM packets

STAM Packet	Acronym	STAM packet Type field
Penalize STAM packet	PSP	0001
Detecting STAM Packet	DSP	0010
Rejoin STAM Packet	RSP	0011
Welcome STAM packet	WSP	0100



(a) Heartbeat message coupled with STAM packet
(encrypted using the GCK of the sender)



ID_{SRC}: id of the sender.
ID_{DST}: id of the receiver.
Counter: the counter of the sender, used to defeat replay attacks
Length: the length of the packet (in number of words)
S_1: current state of the sender
Type: type of the packet
S_2: the target state that the sender informs the receiver to transit to.
Reserved: must be 0.
ID_{TARGET}: the target node to be detected.

(b) STAM packet

Figure 8. STAM Packet

3.2. The Finite State Model Approach

The proposed scheme is based on a finite state model. At any time, a node is in one of the following four states: normal, detecting, penalized, and rejoin. The finite state model is depicted in Figure 9. The state of a node is beared on its own STAM module so that malicious users cannot contaminate it.

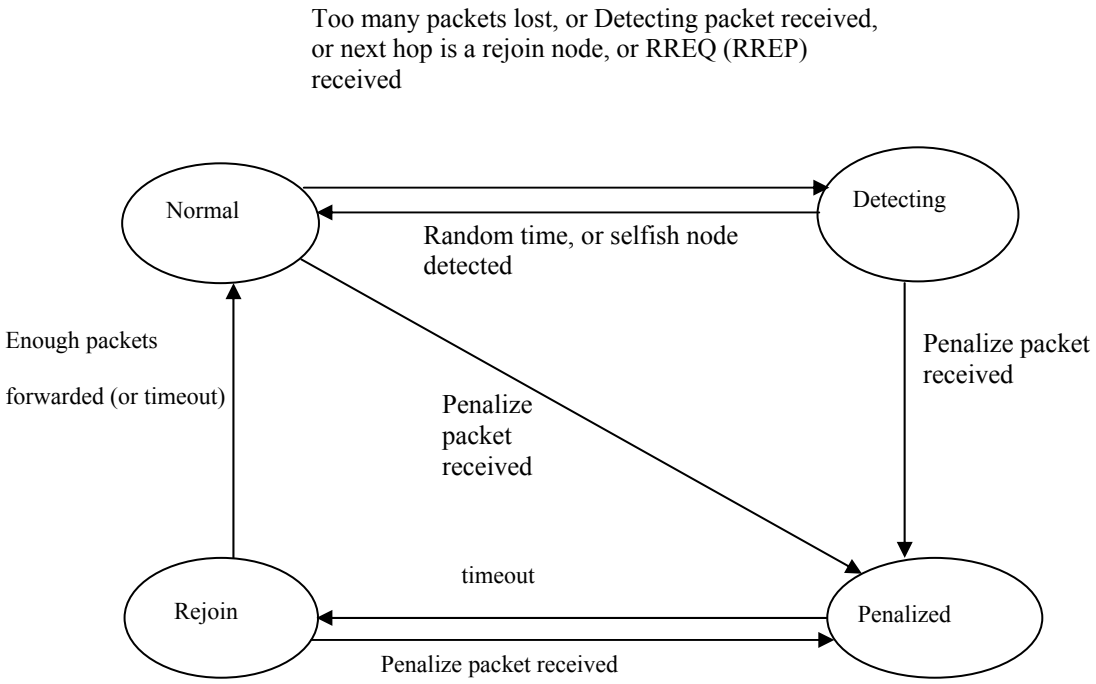


Figure 9. Finite State Model

Before a node can start a data transmission session to another host, it relies on the routing protocol to find a route to the destination. Intermediate nodes can act maliciously and selfishly during the route discovery stage. Hence, nodes are set to the detecting state to identify misbehaving nodes. Once a malicious node is identified, it is switched into the penalized state. The STAM modules "tag" routing packets to help source nodes select only the "clean route" (i.e. a route that does not contain any recognized malicious node) for data transmission. Therefore, after a TCP session begins, the source node assumes that all the intermediate nodes along the adopted route are benign and remains in normal state and no effort is made to detect uncooperative nodes. However, it is possible that some intermediate nodes become uncooperative during data transmission and discard packets that they should relay. In the proposed technique, nodes are switched into the detecting state hop by hop to locate the misbehaving node. Again, once a selfish node is identified, it is switched into the penalized state and will remain in that state for a pre-determined period of time, during which all the other nodes refuse its service requests as a punishment. This way, nodes are given more incentive to act collaboratively. When a node has suffered enough from the penalized state, it will be switched into rejoin state, during which its neighboring nodes are informed and will monitor the rejoining node. If the rejoining node has been observed to provide enough services, it will be turned back to normal state. We elaborate the above process in the following subsections.

3.2.1. Detection Mechanism

We present the proposed detection mechanism in this subsection. We first discuss the attacks we attempt to defeat and then describe the detection mechanism under two different

stages. The detection mechanism can be implemented as a software application as proposed in [7] for lower cost. Alternatively, it can also be implemented as a built-in component of the tamper resistant module for better security. Without loss of generality, we base our discussion on the latter option. We note that users are motivated to cooperate during the detection process since successful identification of misbehaving nodes facilitates their data transmissions.

3.2.1.1. Considered Attacks

The purpose of the STAM scheme is to detect attacks and uncooperative behaviors that result in disruption or degradation of data transmission. We focus on network layer attacks and do not address lower level threats such as physical layer jamming and MAC layer disruptions. The attacks contained by the proposed scheme are as follows. First, the detection mechanism identifies modification, fabrication or selectively dropping of routing packets during route discovery phase. Second, the detection mechanism captures malicious users who deliberately discard data packets that they are obligated to forward either for selfish purposes or to mount denial of service attacks.

3.2.1.2. Route Discovery Misbehavior Detection

During route discovery stage, the STAM module of a node switches itself into detecting state when it receives a RREQ (RREP) packet. We recall that the STAM module intercepts all the packets exchanged between the network layer and the MAC layer. Hence, a STAM module has access to all the incoming and outgoing routing packets and data packets of its hosting node. The STAM module of a detecting node retains each promiscuously learned RREQ (RREP)

packet and the nodes that forward these packets in a buffer. After the detecting node forwards a RREQ (RREP) packet, its STAM module verifies whether a set *P* of neighboring nodes conforms to the routing protocol by checking the recorded routing packets. In DSR and AODV, the set *P* consists of all neighboring nodes since each node should react upon receipt of a RREQ (RREP) packet by transmitting a routing packet at least once. The STAM module of the detecting node determines whether a specific node in *P* has dropped or illegally modified the RREQ (RREP) packet by verifying the retained routing packets. If any of the neighboring nodes are found to be discarding or tampering routing packets, the corresponding misbehavior counter will be incremented. If the MC of a node exceeds a predefined threshold, a Penalize STAM Packet (PSP) (i.e. a STAM packet with its *type* field set to 0001 and S_2 field set to the penalized state) is constructed and will be sent to the misbehaving node. The STAM module of the misbehaving node will turn itself into *penalized* state upon receipt and authentication of the PSP. Once a detecting node has determined the states of all the nodes in *P*, it goes back to the normal state.

3.2.1.3. On-Demand Detection

During a TCP data transmission session, some of the intermediate nodes may become selfish and drop packets. As a result, the transport layer of the source node will have to retransmit the un-acknowledged packets. If the number of un-acknowledged packets exceeds a predefined threshold, the source node of the TCP session invokes the detection mechanism to locate misbehaving nodes.

The detection mechanism works in a hop-by-hop style. At first, the STAM module of the source node is switched to the detecting state and the network interface card of the source node is set to promiscuous mode. We refer to a node that is currently in the detecting state as the "detecting node". The immediate downstream node within the problematic TCP session of a detecting node is referred to as the "target node". The STAM module of the detecting node stores each retransmitted packet. If the target node correctly forwards the packet, the detecting node will "sense" the packet due to the broadcast nature of the media and the STAM module will be able to obtain the packet from the MAC layer and match it with the stored packets; otherwise, if the STAM module fails to receive the packet for a certain amount of time, it increments the corresponding MC and verifies the counter as follows:

1. If MC exceeds a predefined threshold, the target node is identified as a misbehaving node. The STAM module issues a PSP to the target node, which marks itself as in the penalized state upon reception and authentication of the PSP.
4. Otherwise, the target node is benign. In this case, the STAM module of the detecting node generates a Detecting STAM Packet (DSP) (i.e. a STAM packet with its type field set to 0010, S_2 field set to the detecting state and the target id field set to the ID of the next node to be detected) and send it to the target node. The STAM module of the target node will consequently switch itself into the detecting state and trigger the detection mechanism to monitor the node specified in the target id field of the DSP. The current detecting node remains in the detecting state for a random amount of time before it finally returns to the normal state. This is to defeat the attempt of the target node to pretend to be collaborating in aware of the detection mechanism.

The consequence of the above scenarios is that the detecting state is gradually "propagated" from the source node to the intermediate nodes until it reaches the immediate upstream node of the misbehaving host. Eventually either the misbehaving node will be identified and turned into the penalized state or the detecting phase will not identify any node as misbehaving. The latter case can be attributed to two possibilities. First, packet loss is caused by a temporary congestion in the network. No node will be penalized and data transmission will continue on the same route. Second, TCP acknowledgement packets might be sent to the source node through a different path from the one used for data transmission. Hence, packet loss might be due to the existence of malicious nodes along the acknowledgement path. This problem is addressed in a similar way. If TCP acknowledgements are discarded, the source node will have to repeatedly retransmit the unacknowledged packets. Consequently, the destination node will receive duplicate packets and will have to acknowledge them. If a certain number of duplicate packets are received, the destination node will initiate a "reverse" detection to capture the misbehaving node.

Figure 10 depicts the on-demand detection algorithm. Some important features of the detection mechanism are summarized as follows:

1. By adopting an on-demand detection mechanism, the proposed technique can reduce power consumption. Descriptions of energy consumption issues can be found in [83] [89] [14][22].
2. There is no need for a detecting node to report a malicious node to the source node or any other "friend" nodes. As a result, it is not necessary to maintain any complicated trust relationships between hosts as advocated in [1] [5] [6] [52] [53]. The overhead and security vulnerabilities introduced by the scheme are minimized.

3. The fact that a node is malicious is stored on its own STAM module and will be propagated to its neighboring nodes through periodical heartbeat messages. By restricting reputation information within neighborhoods, reputation synchronization can be achieved very efficiently.

```
handlePacket(P, Sender, nextHop)
1.  IF State = Detecting THEN
2.      IF (P is submitted by Network layer) THEN
3.          store(P, buffer[nextHop]);
4.          forward P to the MAC layer;
5.          Start a timer, which calls the Detect function;
5.      ELSE IF (P is promiscuously captured by the MAC layer) THEN
6.          IF (match(P, buffer[Sender]) THEN
7.              remove(P, buffer[Sender]);
8.          END IF
9.      END IF
10  END IF
```

```
Detect(Target)
1.  MC = |buffer[Target]|;
2.  IF (MC > DropThreshold) THEN
3.      sendPSP(Target);  /* Switch Target into the penalized state */
4.  ELSE
5.      sendDSP(Target); /* Switch Target into the detecting state */
6.  END IF;
```

Figure 10. The On-demand Detecting Algorithm

3.3. Malicious Node Punishment and Avoidance

We employ several techniques for benign nodes to avoid and penalize misbehaving nodes. First, a misbehaving node *M* manifests itself to its neighboring nodes through periodical HELLO packets. Consequently, all the neighboring nodes will be able to avoid it. In addition, neighboring nodes will reject each packet originated by *M* as a penalty. We note that in Figure 9 no transition is allowed from penalized state to detecting state. In other words, a node cannot detect and penalize other nodes when it is being penalized. This way, penalized hosts are not able to falsely accuse benign nodes.

Second, we revise the route discovery process. The revision helps benign nodes avoid misbehaving nodes. Given the fact that each routing packet is wrapped by an IP header, we utilize the reserved bit (referred to as the "penalized bit of a RREQ (RREP) packet" hereafter) of the IP Header to ensure that 1) no data is forwarded for a malicious node, 2) the source node is able to avoid a route that contains nodes that are in penalized state, and 3) no modification of routing packets is necessary. We note that a malicious node can selectively discard routing packets. However, this will be captured by the detection mechanism. Therefore, the following discussion is based on the scenario where all the nodes cooperate in forwarding routing packets.

We define a RREQ (RREP) packet whose penalized bit is set as a *penalized RREQ (RREP) packet*. Correspondingly, a RREQ (RREP) packet whose penalized bit is not set is denoted as a *non-penalized RREQ (RREP) packet*. A penalized RREQ (RREP) packet has at least one of the intermediate nodes along its recorded route being recognized as a misbehaving node. With the above definitions we present the following new route discovery mechanism:

1. A RREQ packet formed by the source node is submitted to its STAM module, which checks its current state. If it is currently in the penalized state, the RREQ packet is rejected to penalize the recognized misbehavior of the node. Otherwise, the penalized bit is initialized to 0 and the packet is then flooded in the network.
2. Suppose a node *S* receives a penalized RREQ packet. If *S* has never forwarded any RREQ before, *S* cannot simply discard the packet since otherwise the upstream detecting node will recognize *S* as misbehaving. Instead, *S* submits the RREQ packet to its STAM module, which verifies the recorded state of *S*. If it is in penalized state, it marks the penalized bit of the RREQ packet. Otherwise, nothing is done to the penalized bit and the packet is relayed.
3. If *S* receives a non-penalized RREQ packet, it checks the following two conditions:
    - The same RREQ packet has not been relayed before.
    - *S* has relayed a RREQ before. However, the relayed RREQ packet is a penalized RREQ packet.

    If either of them is satisfied, the RREQ packet is delivered to the STAM module of *S*, which performs the same state verification as discussed above. If neither of the conditions is satisfied, the RREQ packet is ignored.
4. Suppose node *S* is about to send a RREP packet *r*. In other words, *S* has received a RREQ packet *p* and knows of a route to the intended destination (or *S* itself is the destination). In this case, *S* copies the penalized bit from *p* to *r* and delivers *r* to its STAM module, which tags the penalized bit if *S* is in penalized state. We note that a malicious user can deliberately set the penalized bit of r to 0 before *r* is conveyed to

the STAM module. However, such modification will be recognized by the upstream detecting node and will only cause the node to be penalized.

5. The source node receives multiple routes, and will choose the best un-penalized one for data transmission.

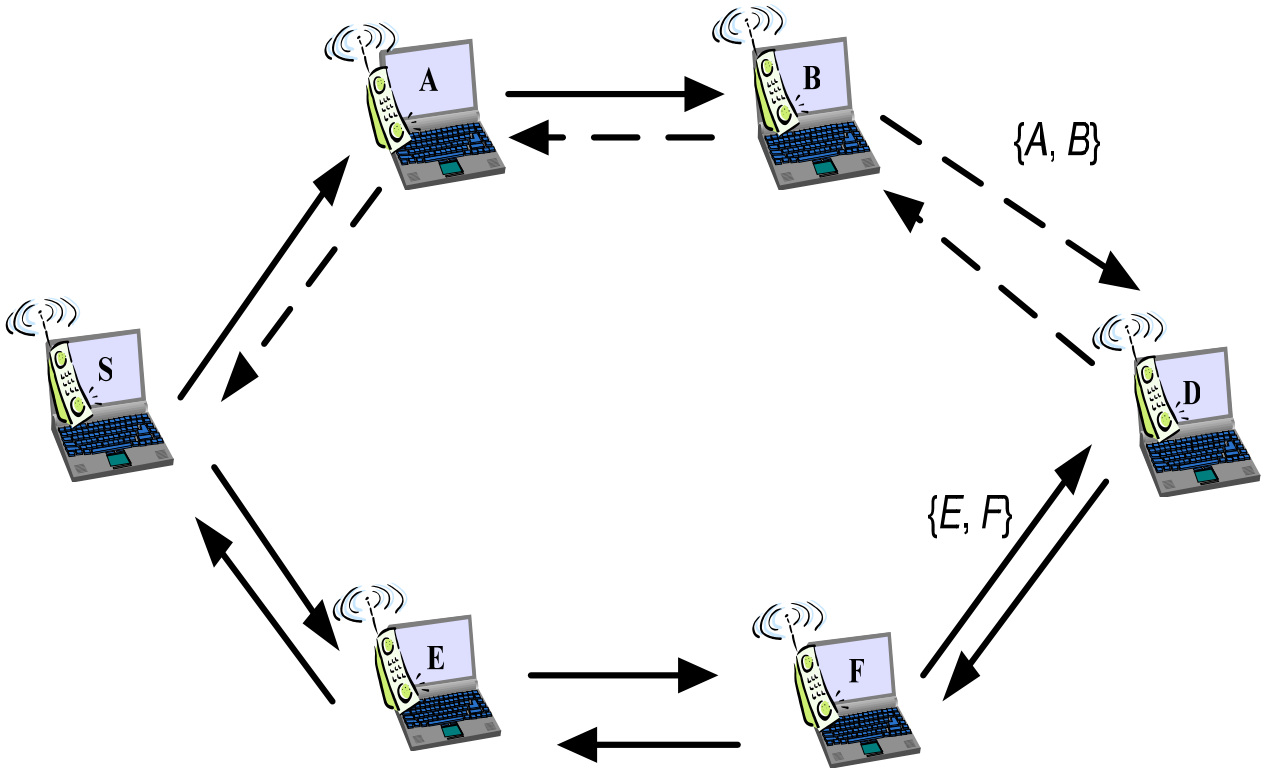


Figure 11. Penalized Bit Based Routing

Figure 11 depicts the misbehaving node avoidance mechanism. Suppose node *B* is a selfish node and it is detected by node *A*. In the STAM technique, node *A* will send a PSP to node *B* and switches *B* into the penalized state. When node *S* initiates a route discovery process, it broadcasts RREQ packets as in DSR. The RREQ will be propagated by intermediate nodes until it reaches node *B*. The STAM module of node *B* will mark the penalize bit of the RREQ before it is relayed to the neighboring nodes. When the destination node *D* receives the RREQ, it creates a RREP and copies the penalize bit from the RREQ to RREP. The RREP will be delivered back to the source node *S*. The same process takes place on node *E* and node *F*. Finally, the source node obtains two routes, {*S*, *A*, *B*, *D*} and {*S*, *E*, *F*, *D*}. Since route {*S*, *A*, *B*, *D*} contains a misbehaving host, node S will choose {*S*, *E*, *F*, *D*} for data transmission.

In Figure 11, the misbehaving node *B* will be penalized. When node *B* intends to send data to some other node, its routing protocol (at the network layer) will construct a RREQ and delivers it to the lower network stack. The RREQ packet will be captured by the STAM module, which realizes that it is in the penalize state and discarded accordingly. Consequently, node *B* will fail to obtain any new route to the destination. On the other hand, any attempt made by node *B* to transmit data using its cached routes will fail as the data packets will be dropped by neighboring nodes as a penalty. In addition, the STAM module of node *B* periodically attaches its recorded state on HELLO messages so that all the neighboring nodes will be able to jointly penalize *B*.

In the proposed approach, malicious nodes can be effectively avoided and penalized through periodical heartbeat packets as well as tallied routing packets. No matter where a malicious node goes, all the nodes in its proximity are able to avoid and penalize it even if they have no a priori knowledge of the node!

3.4. Rejoin of Penalized Nodes

A penalized node is turned into the rejoin state after it has suffered from denial of service for a pre-determined time interval. Once a node is in the rejoin state, it informs all its neighbors through Rejoin STAM packets (i.e. a STAM packet with its type field set to 0011 and S_1 field set to the rejoin state). The neighboring nodes classify the rejoin node according to the Neighbor Classification Update Protocol (NCUP) introduced in the next section. In addition, the STAM module of a rejoin node does not mark the penalized bit of RREQ and RREP packets. Therefore, it becomes possible for a rejoin node to participate in data transmission between other nodes. Each time a benign node *A* transmits a data packet to the next hop *B*[1], the STAM module of A verifies whether *B* is a rejoin node. If yes, *A* is switched into detecting state and monitors *B*'s behavior. Once *B* is observed to have correctly forwarded enough packets, the STAM module of *A* issues a WELCOME packet (i.e. a STAM packet with the *type* field set to 0100 and S_2 field set to *normal* state) to switch the rejoining node to the normal state and conclude the rejoin process. Otherwise, if the rejoining node still discards packets, it will be detected and shifted back to *penalized* state. We note that packets *originated by* a rejoin node will still be rejected by its neighboring nodes. The penalty will be lifted if and only if a rejoin node is observed to be collaborating and is reinstated to normal state.

The rejoin of a malicious node is not fully addressed in most of existing techniques due to the necessity to contact all the nodes to update the reputation of the rejoin node. In our approach, rejoin is accomplished naturally with minimum cost.

---

[1] This is performed by most of the routing protocols and should not be deemed as an overhead incurred by the proposed scheme.

3.5. Combat Evasive Attempts

To avoid the curse of STAM packets, adversaries can modify their nodes to specifically discard or corrupt STAM packets. Our solution is to couple STAM packets with mission critical packets so that discarding or tampering STAM packets will jeopardize data transmission services offered to the offending node. We first present the Neighbor Classification Update Protocol (NCUP) and then discuss the effectiveness of the proposed approach in defeating possible evasive attempts.

3.5.1. Neighbor Classification Update Protocol (NCUP)

The STAM module of a node classifies its neighboring nodes into four types, namely *normal*, *penalized*, *rejoin* and *unreliable* with the assistance of NCUP. NCUP works as follows:

1. When a node *A* encounters a neighboring node *B* through heartbeat messages, it classifies *B* as normal if it correctly receives GCK_B. Otherwise, it classifies *B* as unreliable.
2. When the MAC layer of a node *A* receives a HELLO packet sent by node *B*, it delivers the packet to its STAM module, which processes the packet as follows:
    - The STAM module verifies whether the HELLO message is associated with a STAM packet. If not, the packet is discarded.
    - Otherwise, the STAM module decouples the STAM packet and then decrypts and authenticates using the GCK of the sender. A tampered STAM packet will fail the authentication verification and the entire heartbeat message will be discarded without further processing. We note that symmetric key algorithms are employed

to encrypt and decrypt STAM packets, which makes the whole process very efficient.

- The STAM module classifies node *B* according to the STAM packet as follows:
  a) The S_1 field of a STAM packet indicates the current state of the sender. If S_1 indicates that the sender *B* is in penalized state, *A* classifies *B* as penalized.
  b) If *A* overhears a PSP to penalize *B*, *A* classifies *B* as penalized.
  c) If *A* receives a REJOIN packet from *B*, *A* classifies *B* as rejoin.
  d) Otherwise, *B* is classified as normal.
  e) If *A* either does not receive heartbeat messages from *B* or cannot properly authenticate STAM packets from *B* for more than a predefined number of times, *A* removes *B* from its neighboring node list with all its relevant information.

Periodically, routing protocol of a node updates its route cache according to the types of neighboring nodes. Routes with penalized or unreliable nodes will be removed to avoid misbehaving nodes. Furthermore, a STAM module rejects all outgoing non-routing protocol packets if it is isolated (i.e. it is not aware of any normal neighboring node).

3.5.2. Countermeasures to Evasive Attempts

The possible evasive attempts and their corresponding countermeasures are as follows.

1. A malicious user can attempt to avoid penalty by silently discarding incoming PSPs at the physical or MAC layer. Since all STAM packets are encrypted, the adversary will not be able to distinguish between different types of STAM packets. As a result, a node has to discard *all* incoming STAM packets from *all* its neighboring nodes to avoid being penalized. Eventually, the node will become isolated and will not be able to transmit any data to other nodes.
2. A malicious node can choose to discard outgoing HELLO packets to prevent its state from being propagated to neighboring nodes. This, however, has very limited effect since most of its neighboring nodes have already learned its state by overhearing PSP packets.
3. A malicious node can reset the penalized bit of RREQ (RREP) packets to avoid being excluded by other nodes. For DSR, we addressed this issue by making the STAM module to set the penalized bit of a RREQ (RREP) packet if the embedded route contains penalized neighboring nodes. Since a malicious node generally has no control of its neighboring nodes, clearing penalized bit will not bring any benefit. Similar countermeasures can be derived for other routing protocols.

3.6. Experimental Study

We conducted various experiments to verify the effectiveness of the proposed scheme in enhancing performance of mobile ad hoc networks. In this section, we first introduce the simulation setup and parameters. We then study the proposed technique based on various performance metrics.

3.6.1. Schemes Implemented

We implemented three schemes, namely the reference scheme, the defenseless scheme and the proposed STAM scheme, for performance evaluation. In the reference scheme, all the nodes act collaboratively and relay data for each other. The defenseless scheme was implemented similar to those in [52] and [6]. A certain fraction of nodes are misbehaving as they promise to forward data for other nodes but fail to do so. In other words, these nodes forward routing packets, but discard any data packet not destined at them. No detection or prevention mechanism is implemented so that the network is totally "defenseless". Finally, in the implementation of the STAM technique, a source node of a TCP session is switched into detecting state when at least 2 packets are lost for the particular TCP session. A detecting node S identifies a target node T as a malicious node if at least 50% of the packets forwarded by S to T are lost over a time period of 15 seconds. In our experiments, we set the heartbeat interval τ to be 8 seconds. For fairness, the rejoin mechanism was not invoked in current experiments even though it can further improve network performance.

3.6.2. Simulation Setup

All the experiments were based on GlomoSim [89], a packet-level simulation package for wireless ad hoc networks. The simulations were executed on a Pentium-4 2.5GHz PC with 1GB memory.

Table 2. Simulation Parameters

Parameter	Value
Number of nodes	30
Area	700 meter * 700 meter
Speed	Between 0m/s and 20m/s
Radio Range	250m
Placement	Uniform
Movement	Random waypoint model
MAC	802.11
Sending capacity	2Mbps
Application	FTP
Number of applications	10
Simulation time	10 minutes

Our experiments were based on a mobile ad hoc network with 50 nodes within a 700 by 700 meter two dimensional space. The total simulation duration for each run was 10 minutes (600 seconds). All the nodes employ 802.11 at the MAC layer. At the beginning of each simulation run, nodes were uniformly placed in the area. The random waypoint model was used to model the mobility of hosts. In this model, each node moves in a straight line towards a randomly selected destination location at a speed uniformly distributed between 0 m/s and some maximum speed. After the node reaches the destination location, it pauses for a specified period of time and repeats the aforementioned movement. In our experiments, the maximum speed of a node was limited to 20m/s, resulting in an average speed of 10m/s for each node. We experimented with 0, 5, 10, and 20 malicious nodes, accounting for 0%, 10%, 20% and 40% of total number of nodes respectively. The number of misbehaving nodes is denoted as m. For each value of m, we tested four mobility scenarios, with pause time 0 second, 120 second, 300 second and 600 second. Each configuration was executed under 10 different random seeds and the average values of the metric variables were calculated. FTP was chosen to be the application running on various nodes. The reason of using FTP is because the proposed detection mechanism relies on the TCP protocol. For each simulation run, a total of 10 FTP client/server sessions were generated. The server and client nodes were randomly selected from benign nodes. For each FTP session, 50 randomly generated data packets were transmitted from client to server. Table 1 lists all the simulation parameters.

3.6.3. Metrics

In the experiments, we evaluated the proposed scheme based on the following metrics:

- Network throughput (*T*): we denote the total number of bytes successfully received by FTP server applications as *B* and the simulation time as T_s. Then

  $$T = B / T_s .$$

  This measures the rate at which effective data transmission is performed. It is also a good indicator of the degree of collaboration among the nodes. An undetected misbehaving node would affect the FTP performance, and therefore the overall network throughput.

- Misbehaving node detection ratio (*D*): The ratio between the number of misbehaving nodes that were correctly identified and the total number of misbehaving nodes that have actually acted un-cooperatively during the simulation.

- False accusation ratio (*F*): The ratio between the number of PSP's that incorrectly accused benign hosts and the overall number of PSP's transmitted during the simulation.

- Overhead (*H*): The overhead of the approach is measured as the ratio between the total number of bytes contributed by all the encrypted STAM packets and the total number of bytes contributed by "useful" network layer data (i.e. routing packets and data packets that are successfully delivered to destination nodes) within the network (denoted as *P*). From Figure 8, we observe that the size of a STAM packet is 20 bytes. We denote the total number of nodes as *n* and the size of an encrypted STAM packet as A_S. Suppose the total simulation time is T_s. Each node originates T_s/τ STAM packets throughout the simulation. The overhead can be computed as:

$$H = \frac{T_s \cdot n \cdot A_S}{\tau \cdot P}$$

We set $A_S = 32 bytes$ according to symmetric key encryption techniques such as the Advanced Encryption Standard (AES) **[18]** (256 bit key).

3.6.4. Experimental Results

We present the simulation results of various network configurations in this section.

3.6.4.1. Network Throughput

Figure 12 through Figure 15 depict the network throughput when the number of misbehaving nodes is 0, 5, 10 and 20, respectively. In most of cases, the proposed technique improves network throughput by 20% to 40%.

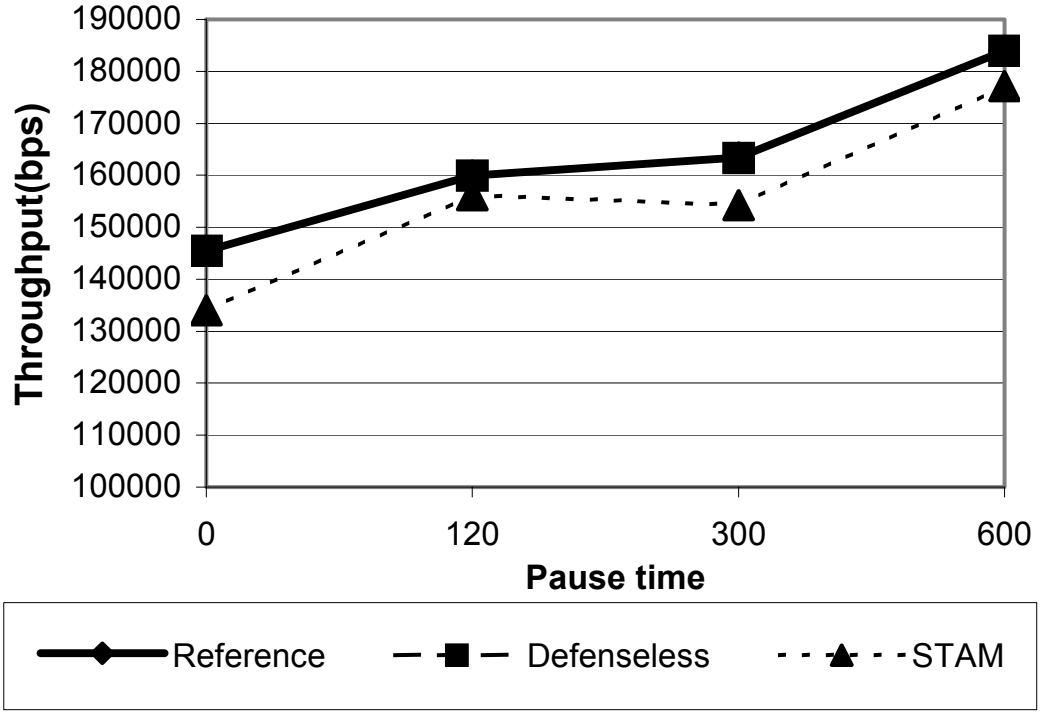


Figure 12. Network throughput for m=0

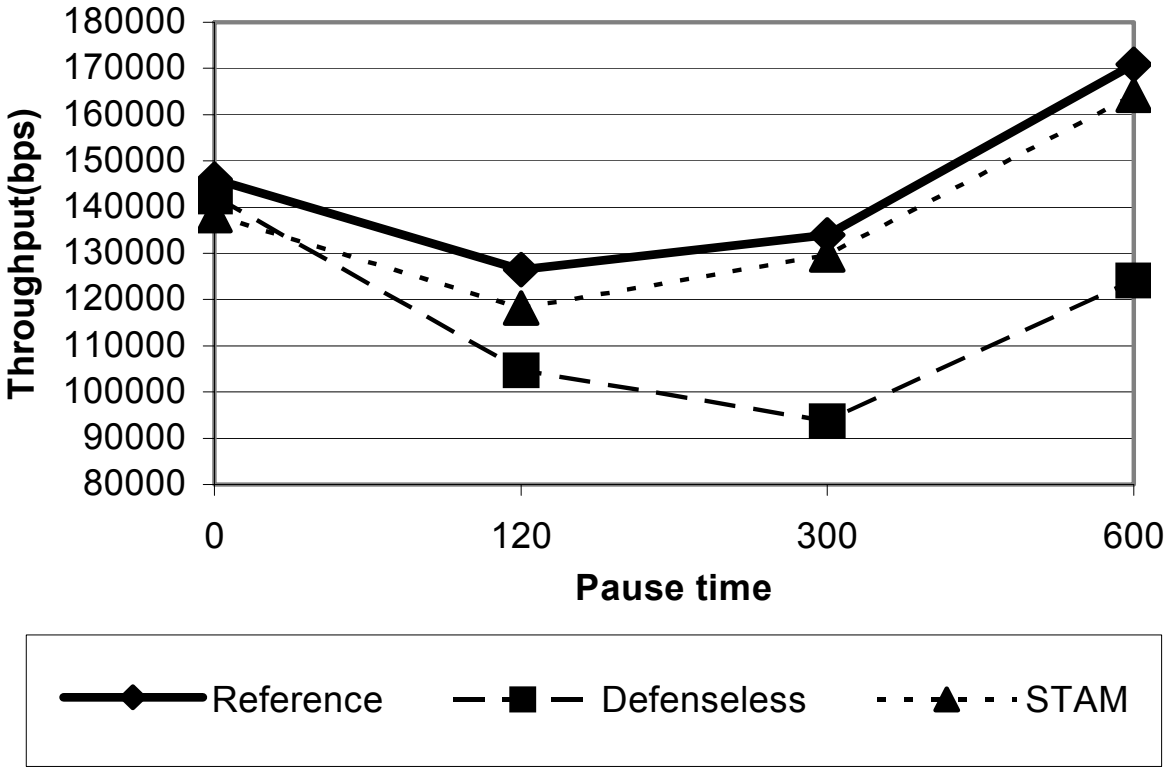


Figure 13. Network throughput for *m*=5

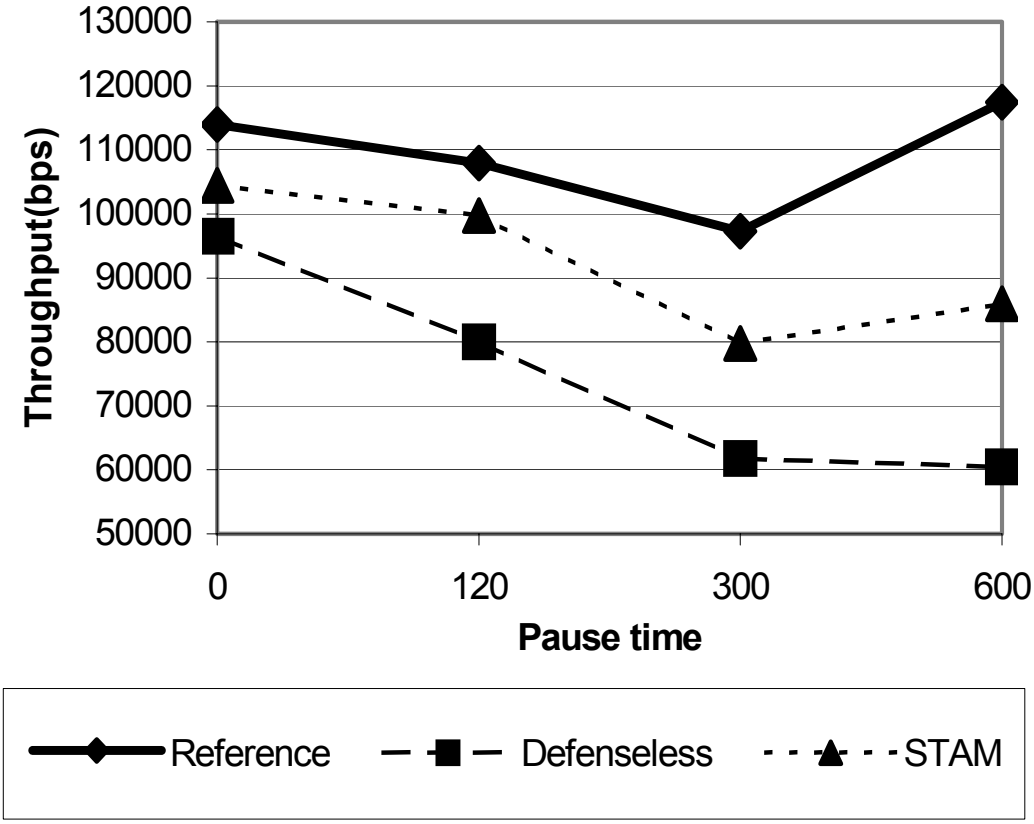


Figure 14. Network throughput for *m*=10

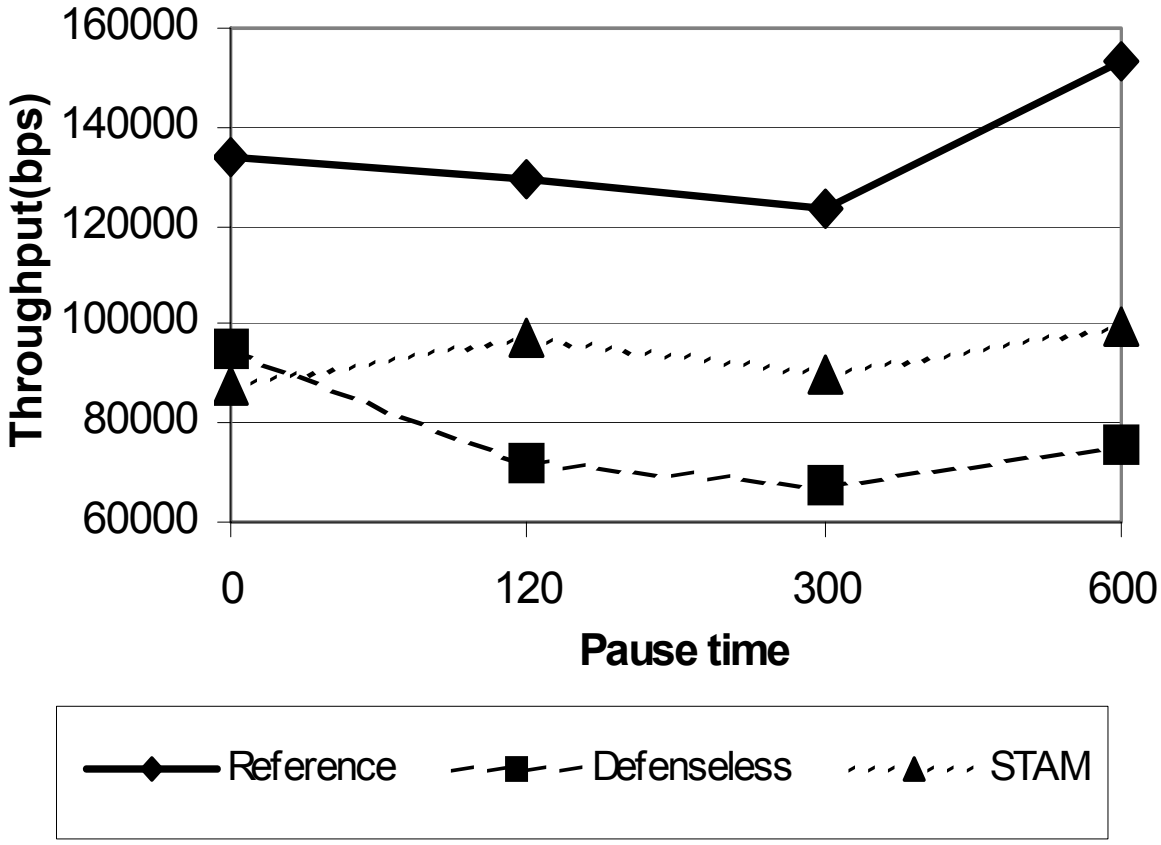


Figure 15. Network throughput for *m*=20

From Figure 13 to Figure 15, we observe that the network throughput in a defenseless network drops significantly with the introduction of misbehaving nodes. For example, in Figure 12 (5 malicious nodes), when the node pause time is 300, the network throughput suffers a 40% degradation. Furthermore, the higher the number of misbehaving nodes, the sharper the network throughput degrades. In Figure 15, when there are 20 (40%) misbehaving nodes and the pause time is 300 seconds, the throughput drops by approximately 60%.

From Figure 12 we notice that when there is no malicious node, the overall performance of a network within which nodes are equipped with our technique is very close to a fully collaborative network. This implies that the proposed approach incurs negligible overhead during normal operation.

By employing the proposed scheme, significantly more data can be successfully delivered to the destinations since nodes always select clean routes for data transmission. As a result, the overall network throughput is greatly enhanced. Figure 13 and Figure 14 depict the practical scenarios where the number of malicious node is 10% and 20% of the total nodes. We observe in most of the cases (pause time greater than or equal to 120 seconds), the system achieves very high throughput improvement. For example, when there are 5 malicious nodes and the pause time is set to 300 seconds and 600 seconds, the throughput of the proposed technique is very close to the reference network. In both cases, the proposed technique improves network throughput by more than 30%. As another example, when there are 10 malicious nodes, the throughput improvement achieved by the proposed scheme is between 25% and 40%. In Figure 15, even in a very unlikely case where 40% of the nodes become malicious, the STAM system still lifted the overall throughput by at least 25% when pause time is greater than or equal to 120

seconds. Such performance enhancement is quite significant since it becomes increasingly difficult for nodes to find clean routes when surrounded by large number of malicious hosts.

Finally, in all the experimental scenarios, when pause time is 0 (i.e. all the nodes are constantly moving), the performance improvement accomplished by the proposed technique is marginal. This is largely due to the high mobility of nodes. In this scenario, frequent link breakage is the dominant factor. Even if clean routes are identified, they quickly become outdated.

3.6.4.2. Misbehaving Node Detection Ratio

We list the results of misbehaving node detection ratio for various simulation scenarios in Table 3. They indicate that the proposed misbehaving node detection mechanism is very effective. In most cases, the detection ratio is about 90%. The results demonstrate that on-demand misbehaving node detection is applicable. Since this approach incurs less energy consumption, it is ideal for MANETs.

Table 3. Detection Ratio

Pause time	0	120	300	600
5 misbehaving nodes	85%	87%	88%	88%
10 misbehaving nodes	87%	90%	88%	85%
20 misbehaving nodes	87%	83%	85%	92%

3.6.4.3. False Accusation Ratio

We report the false accusation ratios of the proposed scheme under various scenarios in Table 4. We conclude that in all node mobility scenarios the false accusation ratio is very low. We observe that this ratio is higher when nodes tend to move a lot. This is due to the fact that some of the benign nodes were forced to drop packets due to link breaks and were thus incorrectly classified by the detection mechanism as misbehaving nodes, thereby lifting the false accusation ratio. Nevertheless, further investigation of simulation log files shows that under all simulation configurations, on average less than one benign node was incorrectly accused. This, in tandem with the detection ratio results presented in the previous sub-section, indicates that the proposed detection mechanism is able to detect most of the in-cooperative nodes with very low false accusation ratio.

Table 4. False Accusation Ratio

Pause time	0	120	300	600
5 misbehaving nodes	6.3%	3.5%	2%	2.4%
10 misbehaving nodes	6.8%	0%	2%	2%
20 misbehaving nodes	5.4%	2.5%	0%	0%

3.6.4.4. Overhead

We present the overhead H of the proposed approach under different number of misbehaving nodes in Figure 17.

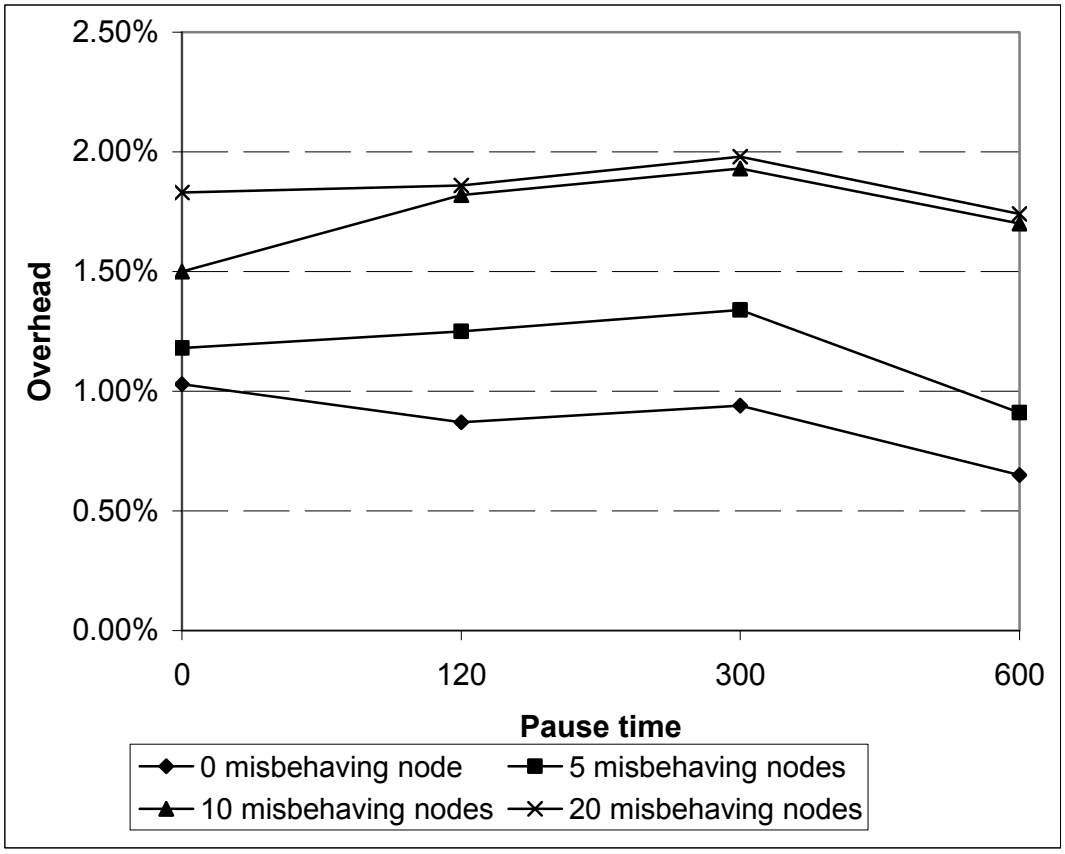


Figure 17. Overhead

We make the following observations. First, in all scenarios, the overhead of our approach never exceeded 2%, essentially negligible. Second, *H* approximately lessened with the decreases in node mobility. We recall that the overhead *H* is defined as the ratio between the number of bytes of encrypted STAM packets and the "useful" data (i.e. overall bytes of both routing packets and data packets that are successfully delivered to their destinations). When nodes tend to be more stable, relatively less routing packets are emitted. However, this is effectively compensated by the increases in the number of data packets successfully delivered to their destination hosts. As a result, the overall number of "useful" data bytes rises and the overhead drops accordingly. Finally, indicates that the overhead rises with the increase in the number of misbehaving nodes. We note that when there are more misbehaving nodes, fewer data packets can make their way to their destinations. Although it generally requires more routing effort to find clean routes, the degradation of data delivery becomes the dominant factor and it lifts the overhead.

CHAPTER FOUR: LOCAL REPUTATION APPROACH

The STAM approach is very efficient and secure in enforcing collaboration of MANET nodes. However, it requires each node to install a security module. In addition, the on-demand detection mechanism is designed to protect applications based on the TCP protocol, but not other transport layer protocols such as UDP. This causes some practical concerns. In this chapter, we introduce a local-reputation approach that addressed all the concerns. Our approach is based on the following fundamental characteristics of MANETs:

- Each packet transmitted by a node A to a destination node more than one hop away must go through one of A's neighboring nodes.
- A's neighboring nodes can overhear its packet transmission.

Given a selfish node *M*, its un-collaborative behavior can be captured by most, if not all, of its neighboring nodes. Each of these nodes will then penalize *M* by rejecting all its packets. As a result, *M* will not be able to send any data to nodes more than one hop away. For a benign node *B*, if *B* is relaying packets for a source node S and is aware that the next hop node *H* is a selfish node, *B* can redirect the packets to avoid *H*. Note that the rerouting operation requires collaboration from *B* for *S*. We also present techniques to enforce such collaboration.

Besides selfish nodes, our technique can also detect malicious nodes mounting denial of service attacks by disrupting link-level packet delivery. Only some of these problems have been studied in the literature [54]. Such attacks are immune to many existing collaboration enforcement techniques such as the Watchdog module proposed in [52] and the CONFIDANT protocol presented in [5][6][7].

4.1. The Detection Mechanism

In this section, we first present the detection mechanism. We illustrate through examples that the proposed detection mechanism can not only identify the second type of selfish behavior (i.e. discarding data packets of other users), but also capture many malicious attacks.

4.1.1. Selfish Node Detection

Each node maintains a list of its neighboring nodes and tracks their actions. Nodes make no assumption of other hosts beyond their direct observable regions. We note that users are motivated to monitor their locality as they will benefit from identifying and circumventing selfish neighboring nodes. Furthermore, our detection mechanism fits naturally into DSR since in DSR nodes constantly sense the media and extract routes from overheard packets.

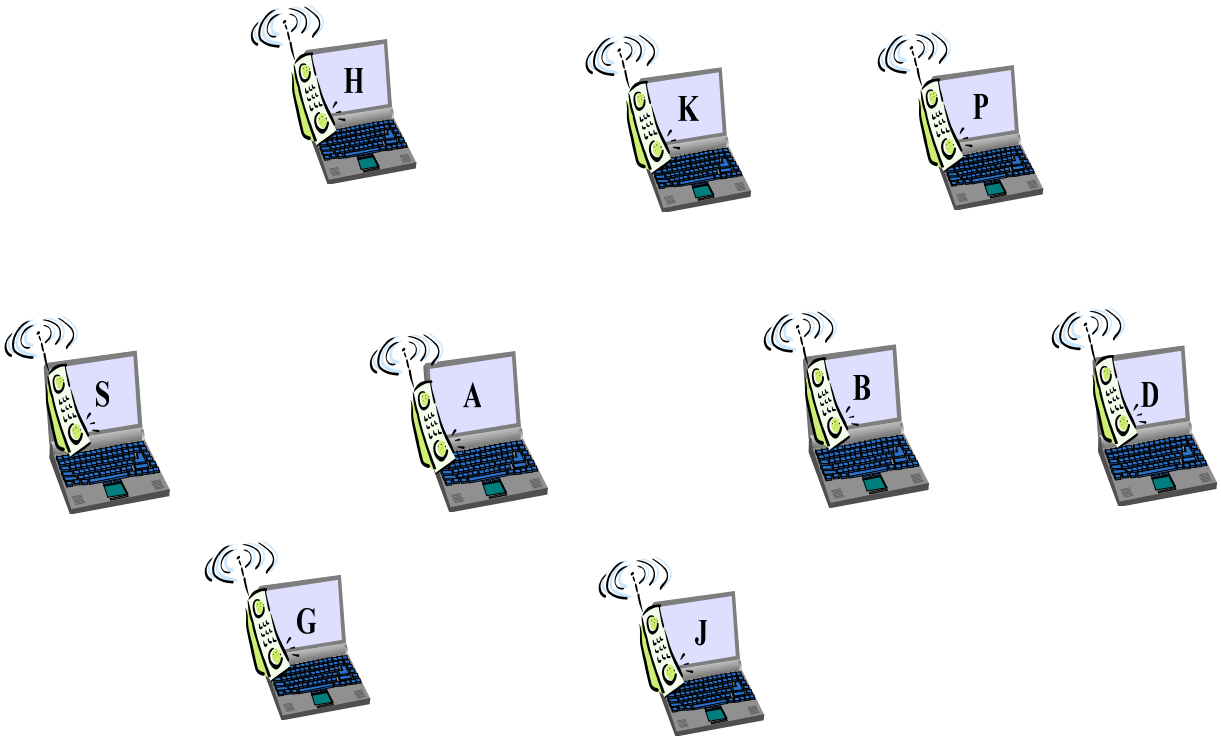


Figure 18. Detection Mechanism

We present the detection mechanism through an example depicted in Figure 18. It shows a node S transmitting data to a node D using a route $\{S, A, B, D\}$. Suppose node A is a selfish node and does not forward data packets to save energy. Assume nodes H and G are neighboring to both S and A, and Nodes K and J are neighboring nodes of both A and B. Each node allocates a memory buffer to store packets transmitted by its neighboring nodes. Let us consider node S first. After S transmits a data packet to A, it

1) records the packet in its local buffer,

2) waits for a certain time interval, and

3) validates whether A has properly forwarded the packet by checking the memory buffer.

Whenever S observes a packet dropped by A (say, at time t), it checks a set Ω of all the packets it has transmitted through node A over a time window defined by $[t - W_{UPPER}, t - W_{LOWER}]$. If the cardinality of Ω is greater than a threshold T_{SUM}, S computes the packet drop ratio for node A on Ω. If this ratio is beyond a given threshold $T_{SELFISH}$, S tallies A as a selfish node; otherwise, S deems A as benign. The purpose of the W_{LOWER} parameter is to make a detecting node ignore packets dropped most recently (perhaps due to link breakage or unexpected network congestion). On the other hand, appropriate W_{UPPER} and T_{SUM} parameters ensure that a detecting node bases its decision on a large enough number of packets and a long enough timeframe. Essentially, selfish intention is sustained if and only if a node has been observed to drop a significant number of packets over a long enough timeframe. With this mechanism, our detection procedure can distinguish link breakage and temporary network congestion from deliberate packet discarding, and effectively reduces false classifications.

In our technique, data transmission is monitored by not only the source and intermediate nodes (i.e, nodes on the selected route), but also their neighboring nodes. Consider nodes *G* and *H* in Figure 18. They, as neighboring nodes of *S*, overhear all data packets sent by *S*. Moreover, both *G* and *H* learn about the next hop (*A* in this example) of each data packet *p* by extracting the source route option field of *p*'s IP header. As *G* and *H* are both neighboring to *A*, they will further detect whether *A* relays the packet using the aforementioned detection technique. In this example, both *G* and *H* will eventually identify *A* as a selfish node based on their own observations. On the other hand, although *K* and *J* are also neighbors of node *A*, they will not be able to detect *A*'s misbehavior since they have no access to the packets sent by the previous hop to *A* (*S* in this example). We refer to this scenario as "*asymmetric sensing*". Our experimental results show that the effect of asymmetric sensing is limited. In most of cases selfish nodes suffer much lower performance than benign nodes. Figure 19 illustrates the detecting algorithm.

HandlePacket(*P*, *Sender*, *nextHop*)

1. IF (*Sender* = Myself) THEN
2. *buffer*[*nextHop*][*P*].*matched* = FALSE;
3. Start a timer, which invokes the *Detect*(*P*, *nextHop*) function;
4. ELSE
5. IF (*match*(*P*, *buffer*[*Sender*]) THEN
6. *buffer*[*Sender*][*P*].*matched* = TRUE;
7. END IF
8. END IF

Detect(*P*, *Target*)

1. IF (*buffer*[*Target*][*P*] = FALSE) THEN
2. Ω = *getPackets*(*buffer*[*Target*], $T\text{-}W_{upper}$, $T\text{-}W_{lower}$);
3. IF ($|\Omega| \geq T_{SUM}$ AND *dropRatio*(Ω) > *dropThreshold*) THEN
4. *Mark*(*Target*, *MALICIOUS*);
5. END IF
6. END IF

Figure 19. Detection Algorithm

4.1.2. Denial of Service Attack Detection

The above mechanism can also detect denial of service attacks mounted by malicious nodes using techniques discussed in [52]. In Figure 18, a malicious node *A* does relay data packets. However, it either controls its transmission power to prevent data packets from reaching its next hop, *B*, or intentionally causes collisions at node *B* to achieve the same effect. In either case, nodes *S*, *G*, and *H* will consider node *A* as a collaborative node whereas *B* never successfully receives any packet. The watchdog approach [52] fails under these situations. In our approach, however, nodes *K* and *J* can detect such attacks by examining the MAC-layer frames. In 802.11, the MAC layer of a node acknowledges the sender for each data frame successfully received. In our example, nodes *K* and *J* will not observe acknowledgement frames from *B* and will thus mark *A* as malicious instead of falsely accusing *B*. We note that malicious users can exploit this mechanism to cause false penalties. In Figure 18, suppose node *A* is benign and node *B* is malicious. Node *B* intentionally refrains from acknowledging packets received from node *A*, hoping to trick neighboring nodes such as *J* and *K* to falsely recognize node *A* as a selfish node. A key observation to defeat such attacks is that a collaborative node *A* will retransmit the pending packets if it does not receive acknowledgements from *B*; whereas no retransmission attempts will be made by a selfish node. Thus, by verifying whether a node conforms to the MAC layer protocol, we can successfully avoid false accusation of node *A*.

4.1.3. Collusion

We compare our technique with existing techniques regarding collusion robustness. In money-incentive models, significant effort needs to be invested (i.e. tamper-proof module) to prevent participants from gaining monetary benefit through colluding. In reputation-based schemes, colluding is attractive to both selfish and malicious users. On one hand, colluding selfish users can successfully cover each other and escape penalty. On the other hand, malicious participants can collaboratively cause various undesirable effects to benign users. In our technique, each node determines the reputation of its neighboring nodes through first-hand experiences, not through "rumor" or "propagated information". As a result, colluding becomes much harder in this new environment.

4.1.4. IP/MAC Address Spoof Detection

A more sophisticated malicious node might seek to spoof its own IP address and/or MAC address to impersonate a neighboring node to either bypass the detection mechanism or cause false penalty. Such address spoofing can be detected by considering the sequence control values of the MAC frames, as pointed out in [86]. The basic idea is that each node in the network keeps track of i) the MAC addresses and ii) the sequence control field of the 802.11 frames sent by all its neighboring nodes. We assume that adversaries are not able to compromise the firmware of network interface cards to manipulate the sequence control field.

A malicious node *M* can instantiate the following attacks. First, it spoofs its own IP/MAC address to mimic a benign node (say node *A*) and drop data packets. The purpose is to cause neighboring nodes to falsely accuse *A*. Suppose the latest sequence control number of *A* is

c_A and the latest sequence control number of M is c_M. We note that in general $c_A \neq c_M$. c_A is maintained on most of A's neighboring nodes. Since M cannot manipulate its own firmware, it cannot put the correct sequence control number in its own data frames. Thus, neighboring nodes of A will not be tricked to penalize A. Second, M might want to spoof its IP/MAC address to circumvent the penalty imposed on it. With the above technique, we can detect either IP address or MAC address spoofing. However, if M changes both its IP and MAC address, it can successfully escape penalty. It is very difficult to detect such behavior. Nevertheless, if M continues its selfish/malicious behavior, it will soon be captured again. Consequently, M has to keep changing its IP and MAC addresses. This might result in losses of session data and failure of the applications running on M.

4.2. The Penalty Mechanism

Punishment of selfish/malicious nodes is achieved as follows. A node dedicates a *detection_time* field for each of its neighboring nodes. Suppose a node *H* identifies a selfish or malicious node *A* at time *t*. It records *t* in the *detection_time* field corresponding to *A*. Meanwhile, *H* keeps monitoring *A* and updates the *detection_time* field if *A* does not cease its misbehavior. *H* drops packets ***originated*** by *A* as a penalty. More specifically, *H*'s decision on whether to forward a data packet *p* for node *A* is based on

$\Delta = t_p$ - *A.detection_time*; where t_p is the time *H* receives *p* and *A.detection_time* is the *detection_time* field corresponding to node *A* on *H*.

If Δ falls within a threshold defined as *penalty interval* τ, *H* will reject the packet. Consequently, the penalty will last as long as *A* continues to misbehave. In other words, the actual penalty time is proportional to the length of *A*'s misbehavior.

One concern of the penalty mechanism is that a benign node might be misclassified when it is penalizing its neighboring misbehaving nodes. We address the problem by slightly modifying the detection mechanism. In particular, a detecting node does not count packets dropped by its neighboring nodes due to selfish node penalty. In Figure 18, suppose node *A* is a selfish node and it discards data packets from node *S*. As explained before, *A* will be detected by nodes *S*, *H*, and *G*. Consequently, node *H* and node *G* will not penalize node *S* when *S* rejects packets originated by node *A* and vice versa. Moreover, we recall that in the detection mechanism, a detecting node forms its decision based on a long enough time window and

sufficient packet count. Since a benign node always relays packets for other (benign) nodes, it is unlikely that its packet drop ratio within a reasonable time window will exceed the $T_{SELFISH}$ threshold. Therefore, the chance of false accusation is slim. Our experimental results also confirm that benign nodes in general do not suffer from false penalties.

4.3. Dynamic Redirection

In reputation-based mechanism, two scenarios will cause a source node to reroute data packets over a particular node. First, when an intermediate node detects a selfish or malicious node, it informs other nodes (including the source node of the session) through *reputation packets* so that they can choose a "clean" route to circumvent the selfish node. Second, Route Error (RERR) packets are transmitted to the source node when broken links are encountered[2]. In both cases, source nodes are responsible for rerouting the data. In the STAM approach, the RREQ and RREP packets are tagged to help the source node avoid misbehaving nodes. In the proposed technique, we allow neither reputation packets nor RERR packets to be propagated. An obvious question is: who should reroute the data packets to bypass both irresponsible nodes and broken links?

[2] Selfish nodes can falsely claim broken links in order to be excluded from packet transmission sessions.

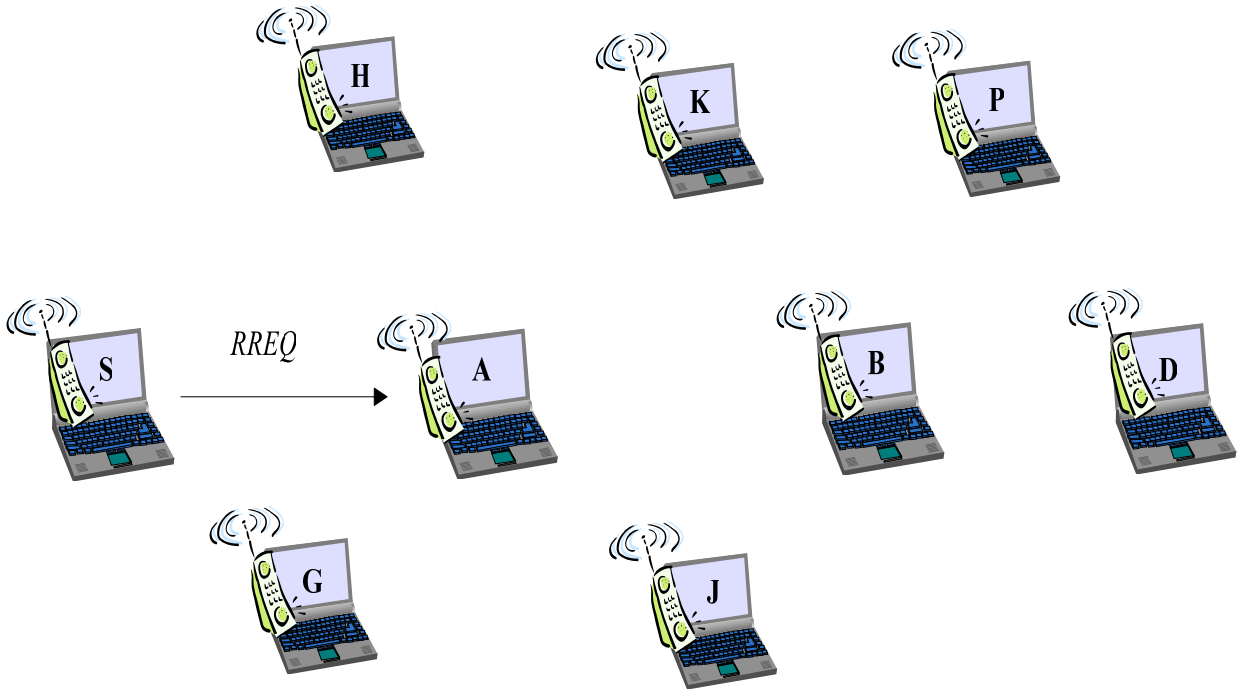


Figure 20. Adaptive Redirection

Our solution is that each node shares the responsibility of rerouting packets. We use Figure 20 to illustrate the idea. We assume that node *S* is sending data to node *D* through a path {*S*, *A*, *B*, *D*}. Suppose the link between node *A* and node *B* is a *malfunction link* (i.e. either broken or node *B* is selfish). Without loss of generality, we assume that node *B* is a selfish node. After relaying a certain number of packets, node *A* will realize that *B* is a selfish node. We refer to node *A* as a *proxy* of source node *S*[3]. In our approach, *A* first purges all paths containing node *B* as an intermediate node from its route cache. Next, when *A* receives subsequent data packets from *S*, it broadcasts a Route Redirect (RRDIR) packet, indicating node *B* as a *bypassing target*. We note that a RRDIR packet serves as an indication of the beginning of the reroute process and the target node to be bypassed. It is by no means a reputation broadcast. In other words, neighboring nodes will not update their views of other nodes based on the RRDIR packets they receive. Continue the above discussion, the proxy node *A* then reroutes the packets by obtaining an alternative clean route to node *D* from its route cache. If such a route does not exist in its cache, *A* will buffer the data packets and instantiate a route discovery process to locate a clean path to *D*. In Figure 1, *A* will discover a new clean route {*K*, *P*, *D*}, revise the embedded route of each data packet and relay them to the destination. In this case, the actual route data packets traverse from *S* to *D* is {*S*, *A*, *K*, *P*, *D*}. It is possible for several proxy nodes to adaptively reroute data packets to avoid multiple selfish nodes along the chosen route. If *A* cannot find a route to *D* after a certain number of retries, it informs *S* through a RERR packet.

The proper functioning of the proposed selfish and malicious node circumvention scheme relies on the collaboration of proxy nodes. Unfortunately, proxy nodes can act maliciously to

[3] The proxy of a source node can be the source node itself when its next hop is selfish.

either avoid the reroute task or mount denial of service attacks. Continue the above example. When node *A* receives a data packet from *S*, it has the following options.

- Node *A* can mount a denial of service attack to *S* by deliberately forwarding packets to *B* even though it is aware that *B* is a selfish node. Nodes *K* and *J* will detect such attack as follows. First, both nodes will identify node *B* as a misbehaving node and they will assume that *A* has reached the same conclusion. Next, as *A* makes no effort to bypass *B*, both *K* and *J* will mark *A* as a malicious node and starts to penalize it.
- *A* does not reroute the packet and simply reports a RERR back to the source. In this case, all its neighboring nodes (*S*, *G*, *H*, *J*, and *K*) hear the RERR packets whereas none of them is aware of any route discovery attempt made by *A*. Thus, all of them will deem *A* as a selfish node.
- *A* broadcasts a RRDIR packet and then starts a route discovery process. Nevertheless, *A* reports a RERR to the source regardless of whether it receives RREP packets from the destination. The countermeasure we design involves utilizing some context information. After *A* sends a RREQ packet to look for a route to *D*, all its neighboring nodes will wait for the RREP packet to come back. Suppose node *K* relays the replying RREP packet to *A* and assume node *H* also hears the packet. Both *H* and *K* will expect to see node *A* transmit data to node *D*. However, as *A* sends a RERR packet, both nodes will recognize *A* as misbehaving. Furthermore, other neighboring nodes (*S*, *G*, and *J*) will deduct certain number of points for node *A* (say, equivalent to one third of those deducted for packet dropping). In other words, failure to reroute data packets is deemed as low-weight misbehavior. The purpose of this design is to discourage un-collaborative behavior. Benign nodes always relay data packets and will not suffer from such deduction.

- *A* broadcasts a RRDIR packet and reroutes data through a fabricated path. This attack has very limited effect in that benign nodes along the faked route will reroute the data packets and node *A* still has to relay data.

A last concern is that malicious nodes might attempt to disrupt data transmission by rerouting data packets. For instance, in Figure 20, suppose *A* is a malicious node. When it receives a data packet from *S* that it should forward to a benign node *B*, it redirects the packet to a different (fabricated) route, hoping that other nodes along the redirected route will drop the packet. We note that this problem also exists in other schemes and is not introduced by our technique. More importantly, our technique facilitates the detection of such attacks. With our redirection mechanism, *A* has to broadcast a RRDIR packet to announce the rerouting operation. Otherwise its neighboring nodes (*S*, *H*, and *G*) will identify it as a malicious node. In the RRDIR packet, *A* has to declare the correct next hop (*B* in this case) that it intends to bypass. Otherwise, it will be captured by *S*, *H*, and *G*. After receiving *A*'s RRDIR packet, node *B* will be aware of *A*'s attempt to deviate packets from a valid route and penalize *A*. Nodes *K* and *J* will also penalize *A* as they both recognize *B* as a benign node through their own observations. Finally, nodes that reroute packets for an excessive number of sessions within a certain time period will be considered as malicious and penalized by their neighbors.

4.4. Experimental Results

We conducted various experiments to evaluate the effectiveness of the proposed technique in enforcing collaboration for MANETs. In this section, we first introduce the simulation setup and parameters. We then discuss the proposed technique based on various performance metrics.

4.4.1. Schemes Implemented

We implemented four schemes, namely the reference scheme, the defenseless scheme, the reputation-based scheme and the proposed experience-based scheme, for performance evaluation. In the reference scheme, all the nodes act collaboratively and relay data for each other. The defenseless scheme was implemented similar to those in [52] and [6]. A certain fraction of nodes are selfish as they promise to forward data for other nodes but fail to do so. In other words, these nodes forward routing packets, but discard any data packet not destined at them. No detection or prevention mechanism is implemented so that the network is totally "defenseless." Next, we implemented a reputation-based system. In this scheme, each node maintains global reputation of other nodes. Nodes update reputation of others as follows. First, nodes monitor and form their opinion about the reputation of neighboring nodes using the same detection mechanism as presented in Section 3.4. Nodes always trust their first-hand experiences with other nodes and ignore any reputation information against their own belief. Next, when a node detects a selfish node, it informs the source node of the communication session through a reputation packet. In response, the source node selects a "clean" route to transmit the remaining

data if necessary. Nodes also update reputation of other nodes based on promiscuously learned reputation packets. Finally, each node periodically broadcasts reputation of other nodes in its locality. We implemented three types of nodes in this scheme, namely benign node, selfish node, and cheating node. A benign node always truthfully broadcasts the reputation information it has observed first hand, and honestly forwards the reputation information from neighboring nodes. A selfish node does not participate in data packet forwarding but cooperates in disseminating reputation information (i.e. it generates and relays reputation packets and never lies about other nodes). A cheating node relays both data and reputation packets for others. During reputation broadcast, however, it always lies about the reputation of nodes that it has direct experiences with. For all other nodes it is aware of, the cheating node simply reports them as selfish.

4.4.2. Simulation Setup

All the experiments were based on GlomoSim [89], a packet-level simulation package for wireless ad hoc networks. The simulations were run on a Pentium-4 2.5GHz PC with 1GB of memory.

Table 5. Fixed Detection Parameters

Parameter	Value
T_{SUM}	8 packets
$T_{SELFISH}$	0.8
Penalty interval τ	180 seconds
Detection buffer size	2MB

Our experiments were based on a MANET of 50 nodes within a 700x700-square-meter 2-dimensional space. The simulation duration for each run was 10 minutes. All the nodes employ 802.11[101] at the MAC layer. At the beginning of each simulation run, nodes were uniformly placed in the area. The random waypoint model was used to model host mobility. In this model, each node moves in a straight line towards a randomly selected destination location at a speed uniformly distributed between 0 m/s and some maximum speed. After the node reaches the destination location, it pauses for a specified period of time and then repeats the movement. In our experiments, the maximum speed of a node was limited to 20m/s. We experimented with 0, 5, and 10 selfish nodes, accounting for 0%, 10%, and 20% of total number of nodes, respectively. Selfish nodes are randomly generated for all the simulation schemes. The number

of selfish nodes is denoted as m. For each value of m, we tested two mobility scenarios, with pause times (denoted as p) of 120 second and 300 second, respectively. We employed the selfish node detection algorithm discussed in Section 3.4 for both the proposed scheme and the reputation-based scheme, with different W_{LOWER} and W_{UPPER} values. We picked 0, 4 second, and 8 second for W_{LOWER} and 15 second, 30 second, and 60 second for W_{UPPER}, resulting in a total of 9 different [W_{LOWER}, W_{UPPER}] pairs. Each node allocates a buffer to store packets forwarded by its neighboring nodes in order to detect selfish nodes. A node can handle a maximum of 50 neighboring nodes and for each neighboring node a maximum of 20 packets are stored. The size of an 802.11 frame is limited to around 2KB. Therefore, the size of the detection buffer is about 2MB for each node. Table 1 lists parameters fixed throughout the experiments. We tested the reputation-based system with 0 and 5 randomly selected cheating nodes. In the experiments, the reputation broadcast interval was set to 10 seconds. Each configuration was executed under 5 different random seeds and the average values of the metric variables are reported. Constant Bit Rate (CBR) applications were used in this study. For each simulation run, we randomly generated a total of 10 CBR client/server sessions. In particular, we generated three selfish sessions (i.e. sessions originated by selfish nodes) and seven benign sessions (i.e. sessions started by benign nodes). The data packet size of each CBR session was chosen to be 552 bytes and packet transmission interval was set to 0.2 second. Table 5 lists all the simulation parameters.

Table 6. Simulation Parameters

Parameter	Value
Number of nodes	50
Area	700 meter * 700 meter
Speed	Between 0m/s and 20m/s
Radio Range	250m
Placement	Uniform
Movement	Random waypoint model
MAC	802.11
Sending capacity	2Mbps
Application	CBR
Number of applications	10
Simulation time	10 minutes

4.4.3. Metrics

In the experiments, we evaluated the proposed scheme based on the following metrics:

- Goodput of benign sessions (G_B): For benign sessions, we denote the total number of bytes successfully received by CBR server applications as B_S and the overall bytes sent by CBR client applications as B_C. Then,

$$G_B = B_S / B_C \ .$$

This metric is a good indicator of the degree of collaboration among the nodes. Successful detection and circumvention of selfish nodes will result in significantly higher goodput.

- Goodput of selfish sessions (G_S): For malicious sessions, we denote the overall bytes sent by selfish source nodes as B'_C and the total number of bytes successfully received by the corresponding CBR server nodes as B'_S. Then,

$$G_S = B'_S / B'_C \ .$$

This measures the effectiveness of the proposed technique in terms of penalizing misbehaving nodes. A good collaboration enforcement technique should ensure a low G_S to discourage misbehaviors.

- Communication cost: The communication cost (hereafter also referred to as "cost" in short) of the proposed scheme O_E is calculated as the ratio between the number of all the control packets (i.e., RREQ, RREP, RERR, RRDIR) originated and forwarded by nodes in the network and the total number of data packets successfully delivered to the

destination nodes. More specifically, $O_E = \frac{C_E}{D_E}$, where C_E is the number of control packets originated and forwarded by nodes in the network and D_E is the number of data packets received by destination nodes. Similarly, the communication cost of the reputation-based scheme is computed as $O_R = \frac{C_R}{D_R}$, where C_R is the number of control packets (i.e., RREQ, RREP, RERR, and reputation packets[4]) originated and forwarded by nodes in the network, and D_R is the number of data packets successfully received by destination nodes. We note that the size of a data packet is generally much larger than the size of a control packet. Nevertheless, the ratio measures the average cost it takes the target scheme to successfully transmit a data packet.

[4] The reputation packets include packets originated by a node that detects a misbehaving node and periodical reputation broadcast transmitted by each node.

4.4.4. Experimental Results

We present simulation results of various network configurations in this section. We observe that in general, the goodput of both benign sessions and selfish sessions is not affected by W_{UPPER} whenever W_{LOWER} is fixed. This suggests that under all experimental scenarios, there are always enough packets falling in the detecting timeframe for nodes to detect selfish neighbors. Given this observation, in this section, we only present the average goodput of both benign and selfish sessions for a specific W_{LOWER} value for the proposed scheme. These results are always compared with the *best* performance result pair $<G_B, G_S>$ achieved by the reputation-based scheme using the same detection mechanism, and under the same mobility pattern and number of selfish nodes. More specifically, for a particular configuration of the reputation-based scheme, a performance result pair $\langle G_B, G_S \rangle$ achieved under a $\langle W_{LOWER}, W_{UPPER} \rangle$ is considered better than another performance pair $\langle G'_B, G'_S \rangle$ achieved under another $\langle W'_{LOWER}, W'_{UPPER} \rangle$ iff $G_B - G_S > G'_B - G'_S$. Ties are broken by selecting a $\langle G_B, G_S \rangle$ pair with higher G_B.

In all the figures, we refer to the proposed scheme as "Experience-*lX*", where X represents the W_{LOWER} value and the reputation-based scheme as "*Reputation-cY*", where Y indicates the number of cheating nodes.

4.4.4.1. Benign Session Goodput

Figure 21 and the column “benign” of Figure 22 through Figure 25 depict the goodput of benign sessions when the number of selfish nodes is 0, 5, and 10.

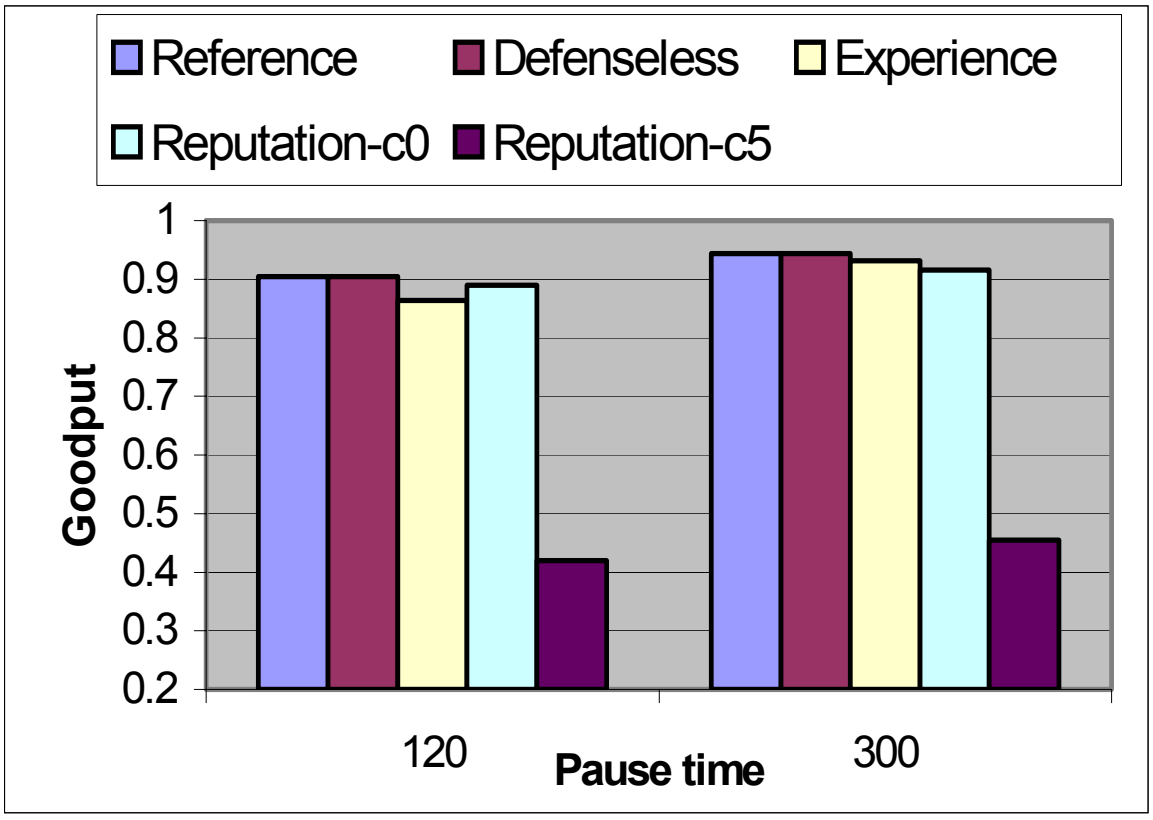


Figure 21. Goodput when m=0

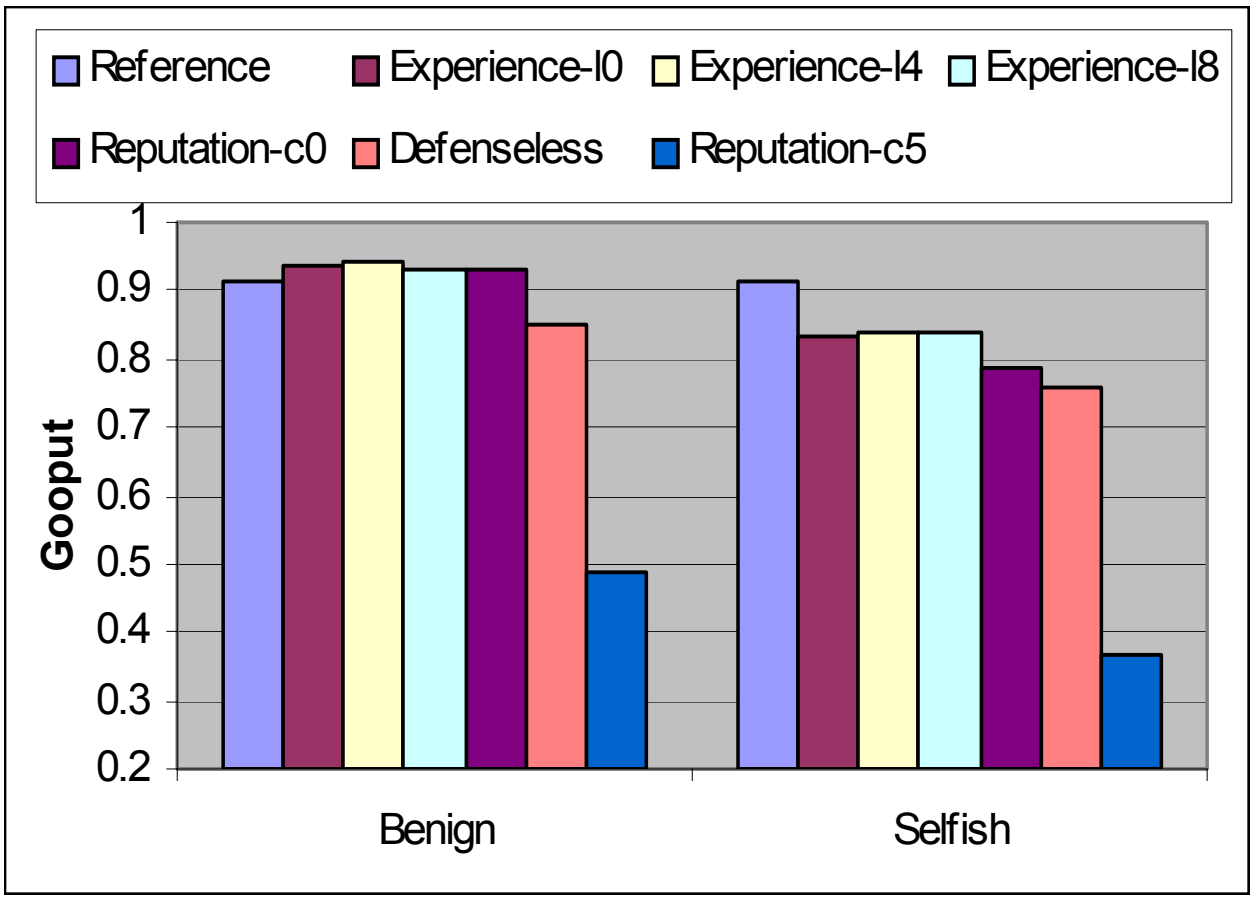


Figure 22. Goodput when p=120, m=5

Figure 21 illustrates the results when there is no selfish node. For the proposed scheme, as results are very similar for all the $\langle W_{LOWER}, W_{UPPER} \rangle$ pairs, we only present the average results. We observe that the overall performance of the proposed technique is very close to that of the fully collaborative network. This implies that the proposed approach incurs negligible overhead.

By employing the proposed scheme, significantly more data are successfully delivered to the destination nodes than the defenseless scheme since proxy nodes proactively detect and reroute data around misbehaving nodes. We can observe this effect in both Figure 22 and Figure 24, where there are 5 selfish nodes. The goodput of the experience-based scheme is always around 0.93 in both scenarios. The improvement over a defenseless network is about 12%. As another example, in Figure 23 and Figure 25, where there are 10 malicious nodes, the proposed technique lifted the goodput from around 0.6 in a defenseless network to higher than 0.85, an improvement of more than 40%. Moreover, the performance is similar under all W_{LOWER} values although $W_{LOWER} = 4$ achieved slightly higher goodput in most of the cases. In general, a lower W_{LOWER} will cause higher false penalties due to temporary link breakage whereas the detection algorithm with a larger W_{LOWER} tends to ignore many of the recently dropped packets and thus unnecessarily delays the reroute and penalty reaction. In addition, the high average goodput confirms that the benign nodes were in general experiencing almost no false accusation caused by penalizing misbehaving nodes.

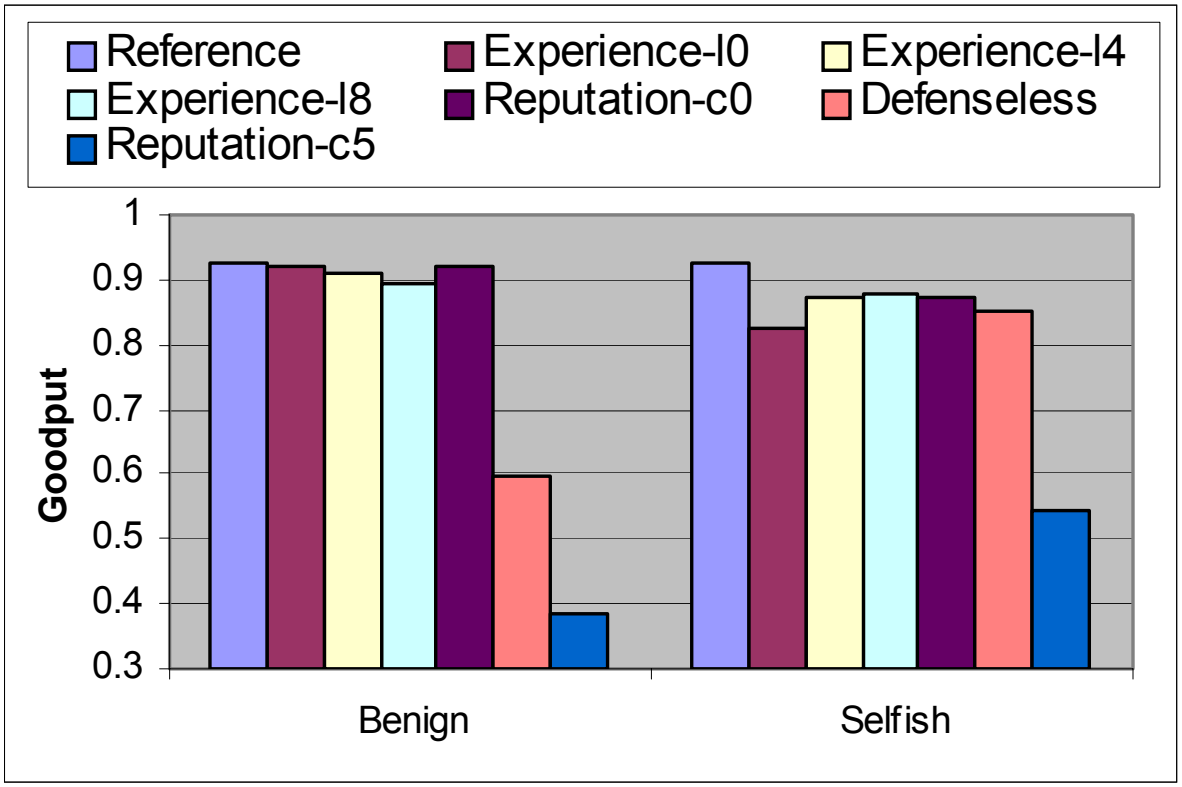


Figure 23. Goodput when p=120, m=10

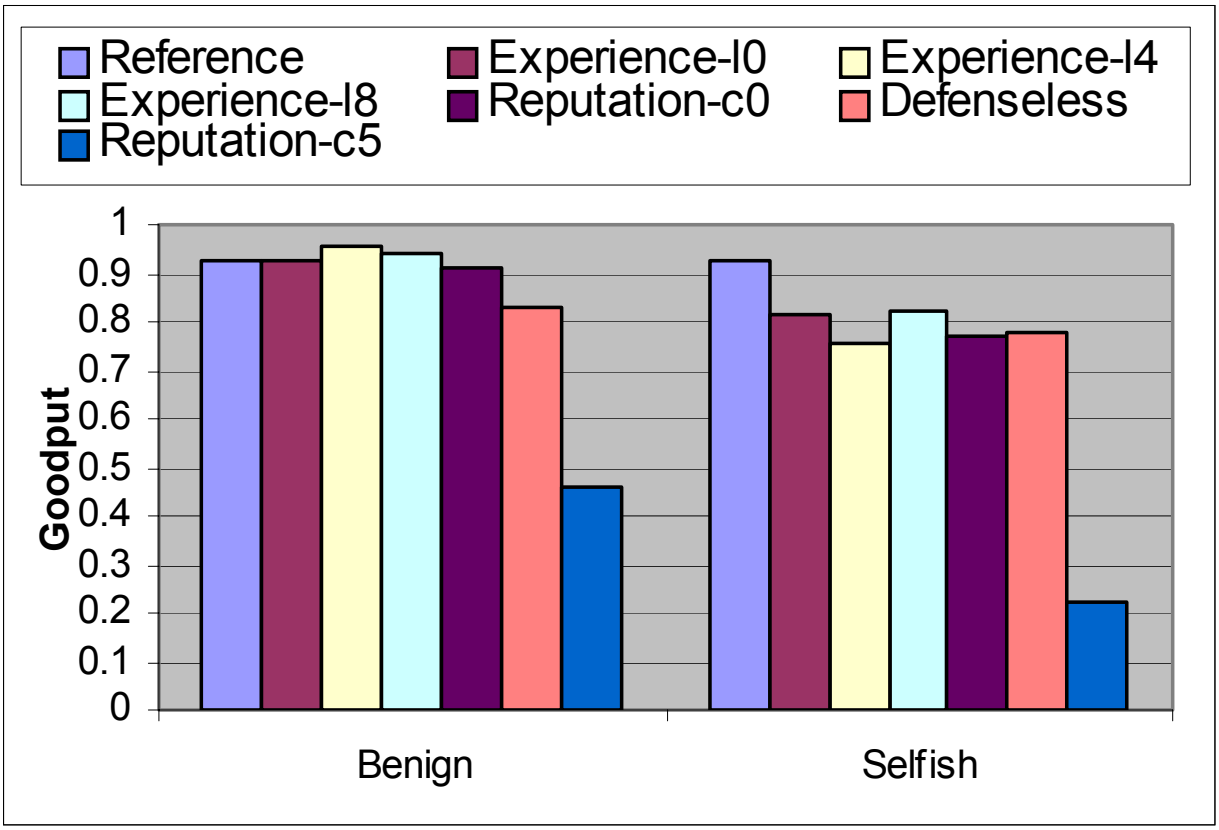


Figure 24. Goodput when p=300, m=5

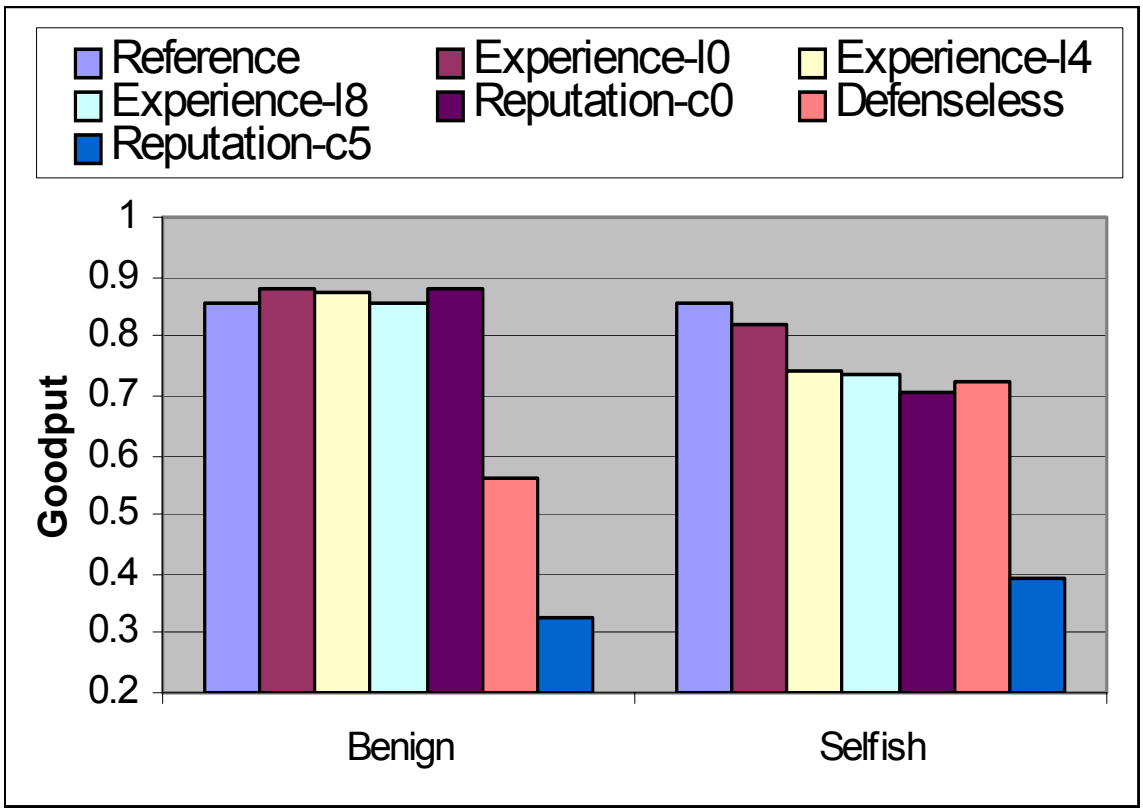


Figure 25. Goodput when p=300, m=10

We also notice from Figure 22 to Figure 25 that the goodput of benign sessions of both the proposed scheme and the liar free reputation-based scheme consistently exceeds the one in a totally collaborative network. Our explanations are as follows. In both approaches, data packets originated by selfish nodes are rejected by their benign neighbors as a penalty. Consequently, such benign neighboring nodes are left with more bandwidth to serve other well-behaved participants, thereby lifting the goodput of benign sessions.

We now compare the performance of the proposed technique with the reputation-based scheme. First, similar performance in terms of benign session goodput is observed for our technique and a liar free reputation-based system. This suggests that prompt packet reroute within the locality of intermediate nodes is in general as efficient as rerouting by source nodes. As a result, reputation propagation becomes unnecessary. In all the experiments, the reputation-based scheme suffered from significant performance loss (more than 50%) when only a few cheating nodes were present. In our simulation, as cheating nodes cooperate in data delivery, they will be deemed as benign nodes and their neighboring nodes will readily accept reputation advertisements from the cheating nodes provided that the recipients have no direct experience with the advertised nodes. As a result, the reputation mechanism was corrupted by inaccurate information and denial of service was experienced by most of the participants. The proposed experience-based approach has none of these problems and is therefore more robust in maintaining good performance.

4.4.4.2. Goodput of Selfish Sessions

We present the simulation results of goodput of selfish sessions in the "selfish" column of Figure 22 to Figure 25.

First, we observe that in most of the cases the goodput of selfish sessions for either the experience-based scheme or the liar free reputation-based scheme is higher than in a completely defenseless configuration. Such improvement is due to the fact that selfish nodes, while not recognized, also detect and actively avoid other uncooperative nodes and therefore also benefit from either the reroute functions in the case of experience-based technique or shared reputation information in the case of reputation-based method.

The experience-based scheme exhibited different behaviors under different W_{LOWER} settings. In general, $W_{LOWER} = 4$ performed better in terms of penalizing selfish participants as it more effectively detects selfish nodes.

In all cases, the goodput experienced by selfish users is lower than what collaborative users enjoy for the experience-based scheme. As an example, in Figure 26, the goodput of benign sessions is higher than 0.93 (left column) as opposed to around 0.81 in the case of selfish sessions. Same phenomenon can be observed in other figures. Thus, selfishness will incur service downgrade and becomes less attractive.

In most of scenarios, the penalty capability of the liar free reputation-based scheme is slightly better than the experience-based approach, as selfish nodes become known to more participating nodes through reputation propagation. We now consider the case when a few cheating nodes exist in the reputation-based system. In practice, cheating nodes will most likely propagate negative reputation of others. As a result, liars actually contribute to the penalty of

selfish nodes since the reputation they propagate with regard to selfish nodes is true. This effect is clearly presented in the experiments. However, such penalty is in the cost of benign nodes. As depicted in Figure 22 to Figure 25, the goodput of benign nodes is significantly hurt. We thus conclude that experience-based scheme is more suitable for MANETs due to its resilience to performance degradation caused by reputation poisoning behaviors.

4.4.4.3. Communication Cost

Figure 26 and Figure 27 illustrate the communication cost of the proposed scheme and the cheat free reputation-based scheme. For our approach, we show the results when $W_{LOWER} = 4$ and $W_{UPPER} = 60$. We also compare results of both schemes with the reference scheme.

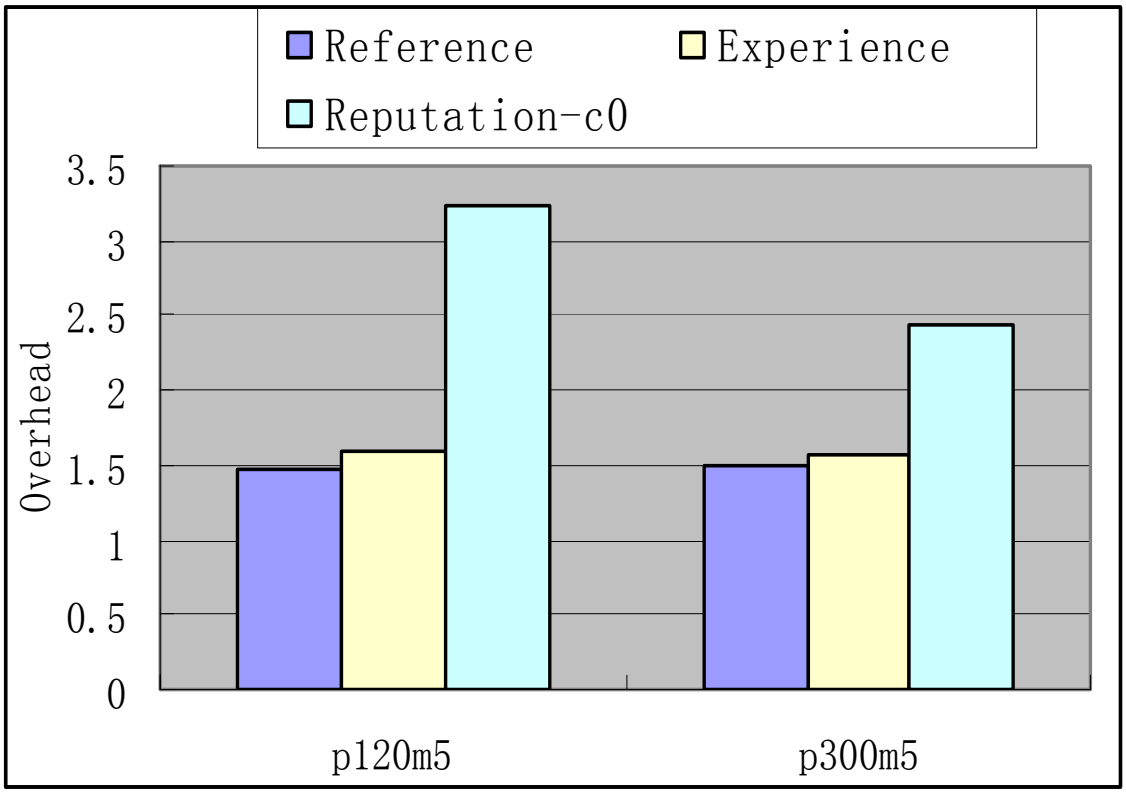


Figure 26. Communication cost when m=5

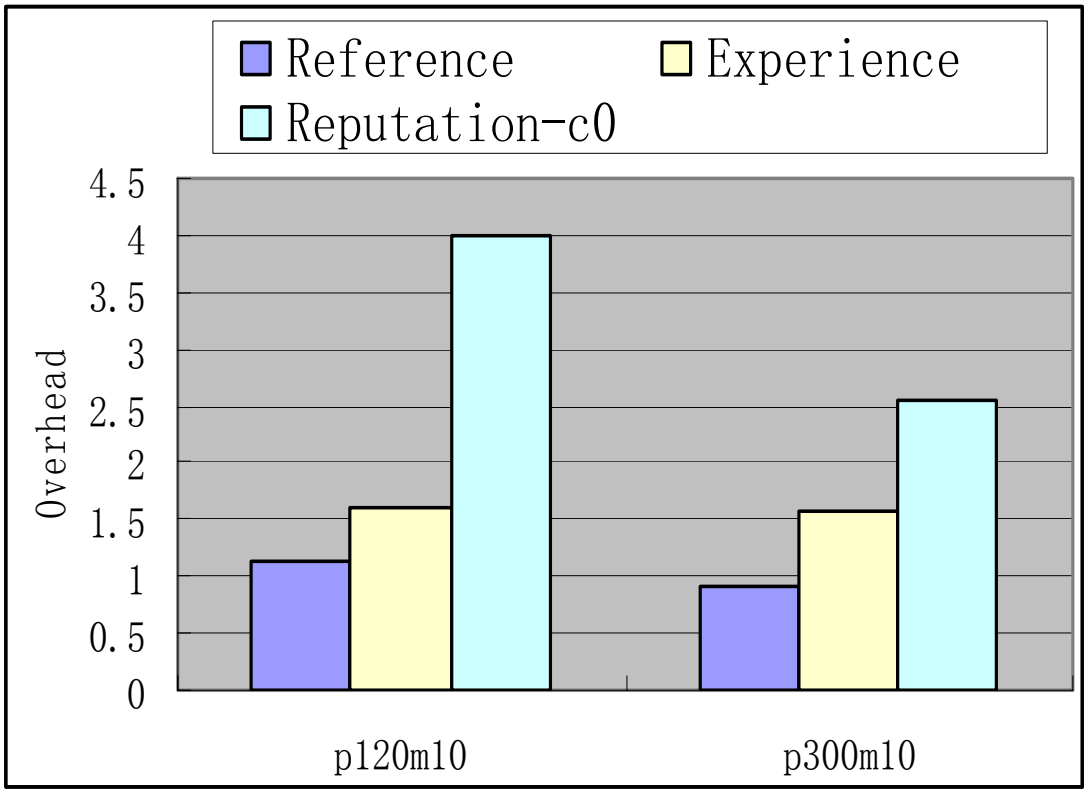


Figure 27. Communication cost when m=10

From the figures we can make the following observations. First, the cost of both our approach and the reputation-based scheme is higher than the cost of a completely attacker free environment. This is because both schemes are aware of misbehaving participants and proactively avoid such nodes, thereby incurring higher routing overhead; whereas the total number of successfully delivered data packets is similar. Second, the cost of the reputation-based mechanism is much higher than our scheme (higher than 67% in most cases). This is because our approach requires no reputation propagation; whereas the reputation-based scheme has to flood reputation information throughout the network. Although both schemes can achieve similar goodput for benign sessions (as illustrated by Figure 21 through Figure 25), our scheme is significantly more scalable and is thus more desirable for MANETs. Next, consider a fixed number of malicious nodes: the lower the node mobility, the lower the cost of both schemes. Obviously, when mobility is low, less routing packets are initiated. On the other hand, more packets are successfully delivered to the destination nodes, hence the lower communication cost. Finally, for a fixed mobility configuration, the higher the number of misbehaving nodes, the higher the communication cost. This also fits the intuition as nodes have to work more diligently when more un-collaborative participants are present.

CHAPTER FIVE: CONCLUSION

In mobile ad hoc networks, there is no fixed infrastructure readily available to relay packets. Instead, nodes are obligated to cooperate in routing and forwarding packets. However, it might be advantageous for some nodes not to collaborate for reasons such as saving power and launching denial of service attacks. Therefore, enforcing collaboration is essential in mobile ad hoc networks.

In most existing techniques, collaboration enforcement is achieved by a detect-and-react mechanism. In which, each node maintains global reputation of others in order to avoid and penalize misbehaving nodes. Propagation of reputation information is accomplished through complicated trust relationships. Such techniques incur scalability problems and are vulnerable to various reputation poisoning attacks.

In this dissertation, we make the following contributions to enforcing collaboration and security in mobile ad hoc networks:

1. We propose a novel approach to protect MANETs against selfish nodes. In our approach, nodes keep local reputation of their neighboring nodes through direct observation. Nodes trust only their first-hand observations and will not be misled by other malicious nodes. In addition, the scalability of MANETs is greatly improved since we do not rely on propagating reputation information throughout the network.
2. We propose two schemes. In the first scheme, a finite state model is designed. Nodes switch between states based on their observed behaviors. In addition, each node is equipped with a tamper-proof module, namely, the STAM module. The STAM module

stores mission-critical information such as public/private keys and node states. It is also responsible for detecting selfish neighbors by investigating packets exchanging between the network and MAC layer. Furthermore, during route discovery, the STAM module marks routes contaminated by misbehaving nodes to assist source nodes to avoid misbehaving nodes. In the second scheme, we do not assume any security module. No reputation advertisement is initiated or accepted. A detection mechanism is provisioned to identify various attacks as well as differentiating selfish behavior from temporary link breakage. We also design an adaptive rerouting mechanism, where nodes dynamically redirect data packets to bypass recognized adversaries in their proximities. The redirect operation is also guarded against various evasive attempts.

The advantages of our approach are many. First, since it does not rely on propagated reputation information, there is no need to maintain complex trust relationships. Second, since the misbehavior detection mechanism is based on first-hand experience at individual nodes, denial of service attacks are much more difficult to achieve. Colluding among nodes to secretly carry out fraudulent actions becomes much more difficult. In the STAM scheme, detection is performed only if it is necessary, thereby improving the energy efficiency of the nodes. In addition, the STAM module carries the state of a node. Therefore, a selfish node will be recognized and avoided instantly. The second scheme does not rely on tamper-proof modules. Nevertheless, it retains most of the advantages of the STAM scheme. With the adaptive redirect mechanism, MANET nodes can bypass misbehaving nodes instantly. It greatly improves network throughput.

We conducted various experiments to investigate the effectiveness and efficiency of the proposed schemes. Simulation results, based on GlomoSim, indicate that both schemes are very

effective in improving network performance. They also work well in disciplining defecting hosts. More importantly, the success of the proposed technique does not rely on reputation exchange and is thus both scalable and robust.

APPENDIX: LIST OF PUBLICATIONS

Journal Publications

[1]. **N. Jiang**, Kien A. Hua, and Mounir Tantaoui, "Collaboration Enforcement and Adaptive Data Redirection in Mobile Ad Hoc Networks Using Only First-Hand Experience," in IFIP International Federation for Information Processing (electronic journal by Springer publisher), Vol. 162/2005, pp. 227-238. (Selected from 2004 IFIP/IEEE International Conference on Mobile and Wireless Communication Networks).

[2]. **N. Jiang**, R. Villafane, Kien A. Hua, A. Sawant, K. Prabhakara, "ADMiRe: An Algebraic Data Mining Approach to System Performance Analysis", IEEE Transactions on Knowledge and Data Engineering (TKDE), pp 888-901, July 2005.

Conference Publications

[1]. **N. Jiang**, Simon Sheu, Kien A. Hua, Onur Ozyer, " A Finite-State-Model Scheme for Efficient Cooperation Enforcement in Mobile Ad Hoc Networks", The 11th IEEE International Conference on Parallel and Distributed Systems, July 20-25, 2005.

[2]. **N. Jiang**, Kien A. Hua, Mounir Tantaoui, "Collaboration Enforcement and Adaptive Data Redirection in Mobile Ad Hoc Networks using only First-Hand Experience", 2004 *IFIP/IEEE International Conference on Mobile and Wireless Communications Networks*, Oct 25-27, 2004.

[3]. **N.Jiang**, Yao H. Ho, Kien A. Hua, "Range Multicast Routers for Large-Scale Deployment of Multimedia Application", *ACM Multimedia* 2004, Oct 11-15, 2004.

[4]. Kien A. Hua, **N. Jiang**, Rui Peng, Mounir Tantaoui, "PSP: A Persistent Streaming Protocol for Transactional Communications", 2004 *IEEE International Conference on Communications, Circuits and Systems*. June 25-28, 2004.

[5]. H. Yu, **N. Jiang**, Annie S. Wu, "Populating Genomes in a Dynamic Grid," *GECCO 2004,* June 27-29, 2004.

[6]. **N. Jiang**, R. Villafane, Kien A. Hua, and D. Tran, "ADMiRe: An Algebraic Approach to System Performance Analysis Using Data Mining Techniques," in *Proc. of ACM SIGAPP Symposium on Applied Computing* (SAC 2003), March 9-12, 2003.

[7]. M. Tantaoui, Kien A. Hua, and **N. Jiang**, "A Limitless Infrastructure for Next-Generation Large-Scale Simulation and Training Systems," in *Proc. of Advanced Simulation Technologies Conference* (ASTC 2003), March 30 – April 3, 2003.

[8]. K. Vu, Kien A. Hua, and **N. Jiang**, "Improving Image Retrieval Effectiveness in Query-by-Example Environment," in *Proc. of ACM SIGAPP Symp. on Applied Computing* (SAC 2003), March 9-12, 2003.

[9]. **N. Jiang**, Kien A. Hua, and J. Oh, "Exploiting Pattern Relationship for Intrusion Detection," in *Proc. of IEEE Symposium on Applications and the Internet* (SAINT 2003), January 27-31, 2003.

[10]. J. Oh, Maruthi Thenneru, and **N. Jiang**, "Hierarchical Video Indexing Based on Changes of Camera and Object Motions," in *Proc. of ACM SIGAPP Symposium on Applied Computing* (SAC 2003), March 9-12, 2003.

[11]. **N. Jiang**, Kien A. Hua, and S. Sheu, "Considering both Intra-Pattern and Inter-Pattern Anomalies in Intrusion Detection," in *Proc. of IEEE Int'l Conf. on Data Mining* (ICDM 2002), December 9-12, 2002.

[12]. **N. Jiang**, Kien A. Hua, Morgan C. Wang. A Parameter Reduction Based Technique for Automatic Analysis of Database Systems. Joint Statistician Meeting 2002.

[13]. Duc A. Tran, Kien A. Hua and **N. Jiang**. "A Generalized Design for Efficiently Broadcasting on Mutiple Physical Channel Air-Cache," in *Proc. of ACM-SIGAPP Annual Symposium on Applied Computing (SAC 2001*), March 11-14, 2001.

LIST OF REFERENCES

[1] B. Awerbuch, D. Holmer, C. Nita-Rotaru and H. Rubens. An On-Demand Secure Routing Protocol Resilient to Byzantine Failures. ACM Workshop on Wireless Security (WiSe) 2002.

[2] R. Axelrod. The Evolution of Cooperation. Basic Books, New York, 1984.

[3] B. Bellur, R. G. Ogier, F. L. Templin. Topology Dissemination Based on Reverse-Path Forwarding (TBRPF) RFC 3684, February 2004.

[4] S. Buchegger and J. L. Boudec, IBM Research Report: The Selfish Node: Increasing Routing Security in Mobile Ad Hoc Networks. RR 3354, 2001.

[5] S. Buchegger and J. L. Boudec. Nodes Bearing Grudges: Towards Routing Security, Fairness, and Robustness in Mobile Ad Hoc Networks. In Proceedings of the Tenth Euromicro Workshop on Parallel, Distributed and Nework-based Processing, pages 403-410, Canary Islands, Spain, January 2002. IEEE Computer Society.

[6] S. Buchegger and J. L. Boudec. Performance Analysis of the CONFIDANT Protocol: Cooperation of Nodes – Fairness In Dynamic Ad Hoc Networks. In Proceedings of IEEE/ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC), Lausanne, CH, June 2002.

[7] S. Buchegger, H. L. Boudec, Coping with False Accusations in Misbehavior Reputation Systems for Mobile Ad-hoc Networks. EPFL Technical Report IC/2003/31.

[8] L. Buttyan and J. Hubaux. Enforcing Service Availability in Mobile Ad Hoc WANs. In Proceedings of IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC), Boston, MA, USA, August 2000.

[9] L. Buttyan and J. Hubaux. Stimulating Cooperation in Self-Organizing Mobile Ad Hoc Networks. Technical Report DSC/2001/046, EPFL-DI-ICA, August 2001.

[10] L. Buttyan, I. Vajda. Towards Provable Security for Ad Hoc Routing Protocols. 2004 ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '04).

[11] D. Câmara, A. A. F. Loureiro. GPSAL (GPS Ant-Like Routing Algorithm) - A Novel Routing Algorithm for Hoc Networks, Baltzer Journal of Telecommunications Systems, 18:1-3, 85-100, Kluwer Academic Publishers, 2001.

[12] Aldar C-F. Chan. Distributed Symmetric Key Management for Mobile Ad Hoc Networks. INFOCOM 2004.

[13] I. Chakeres, E. Belding-Royer AND C. Perkins. Dynamic MANET On-demand Routing Protocol (DYMO), Internet Draft, draft-ietf-manet-dymo-03.txt, work in progress, October 2005.

[14] J-H Chang and L. Tassiulas. Energy Conserving Routing in Wireless Ad-hoc Networks. INFOCOM 2000.

[15] C-C Chiang, H-K Wu, Winston Liu, Mario Gerla. CGSR (Clusterhead Gateway Switch Routing protocol) - Routing in Clustered Multihop, Mobile Wireless Networks with

Fading Channel, IEEE Singapore International Conference on Networks, SICON'97, pp. 197-211, Singapore, 16.-17. April 1997, IEEE.

[16] J. Crowcroft, R. Gibbens, F. Kelly, and S. Ostring. Modelling Incentives for Collaboration in Mobile Ad Hoc Networks. 1st International Symposium on Modelling and Optimization in Mobile Ad Hoc and Wireless Networks (WiOpt'03), 2003.

[17] T. Dierks and C. Allen. The TLS protocol version 1.0. RFC 2246, 1999.

[18] J. Daemen, V. Rijmen. Smart Card Research and Applications, LNCS 1820, J.-J. Quisquater and B. Schneier, Eds., Springer-Verlag, 2000, pp. 288-296.

[19] S. Eidenbenz and L. Anderegg, Ad hoc-VCG: A Truthful and Cost-Efficient Routing Protocol for Mobile Ad Hoc Networks with Selfish Agents. In Proceedings of the Ninth Annual International Conference on Mobile Computing and Networking (MobiCom 2003), September 2003.

[20] E. Gafni, D. Bertsekas. FORP (Flow Oriented Routing Protocol): Distributed Algorithms for Generating Loop-free Routes in Networks with Frequently Changing Topology, IEEE Transactions on Communication, Vol. 29, No. 1, Jan, 1981, pp.11-15.

[21] J.J. Garcia-Luna, M. Spohn. Source-Tree Routing in Wireless Networks, Proceedings of the 7th International Conference on Network Protocols, IEEE ICNP 99, Toronto, Candada, pp. 273-282, IEEE, October 1999.

[22] A. J. Goldsmith, S. B. Wicker. Design Challenges for Energy-Constrained Ad Hoc Wireless Networks. IEEE Wireless Communications, pages 8-27. August 2002/Vol. 9 No.4.

[23] S. Guo, O. W. Yang. Performance of Backup Source Routing (BSR) in mobile ad hoc. In Proceedings of 2002 IEEE Wireless Networking Conference.

[24] M. Günes. ARA: the ant-colony based routing algorithm for manets. In Stephan Olariu, editor, Proceedings of the 2002 ICPP Workshop on Ad Hoc Networks (IWAHN 2002), pages 79-85, IEEE Computer Society Press, August 2002.

[25] Z. J. Haas, M. R. Pearlman, P. Samar. IARP: The Intrazone Routing Protocol (IARP) for Ad Hoc Networks, Internet Draft, http://www.ietf.org/proceedings/02nov/I-D/draft-ietf-manet-zone-iarp-02.txt, July 2002.

[26] Z. J. Haas, M. R. Pearlman, P. Samar. The Zone Routing Protocol (ZRP) for Ad Hoc Networks, Internet Draft, http://www.ietf.org/proceedings/02nov/I-D/draft-ietf-manet-zone-zrp-04.txt, July 2002.

[27] L. Harte, M. Hoenig, D. McLaughlin, R. Kikta. IS-95 CDMA for Cellular and PCS: Technology, Applications, and Resource Guide. McGraw-Hill Professional 1999.

[28] Q. He, D. Wu, P. Khosla. SORI: A Secure and Objective Reputation-based Incentive Scheme for Ad-hoc Networks. 2004 IEEE Wireless Communications and Networking Conference (WCNC 2004).

[29] Y.-C. Hu, A. Perrig, and D.B. Johnson. Ariadne: A secure On-Demand Routing Protocol for Ad Hoc Networks. In Proceedings of the 8th Annual International Conference on Mobile Computing and Networking, ACM Press, 2002.

[30] Y. C. Hu, A. Perrig, and D. B. Johnson. Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks. Technical Report TR01-383, Department of Computer Science, Rice University, December 2001.

[31] Y. C. Hu, D. B. Johnson and A. Perrig. SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless AdHoc Networks. In Proceedings of the 4th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA 2002). IEEE, Calicoon, NY, June 2002.

[32] Y. C. Hu, A. Perrig, and D. B. Johnson. Packet Leashes: A Defense against Wormhole Attacks in Wireless Ad Hoc Networks. In Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003), San Francisco, CA, 2003.

[33] Y. C. Hu and D. B. Johnson. Wormhole Detection in Wireless Ad Hoc Networks. Technical Report TR01-384, Department of Computer Science, Rice University, December 2001.

[34] Y. C. Hu, A. Perrig, and D. B. Johnson. Rushing Attacks and Defense in Wireless Ad Hoc Network Routing Protocols. In Proceedings of the 2003 ACM Workshop on Wireless Security (WiSe 2003), San Diego, CA, September 2003.

[35] Yih-Chun Hu, David B. Johnson. Securing Quality-of-Service Route Discovery in On-Demand Routing for Ad Hoc Networks. 2004 ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '04).

[36] J. Hubaux, L. Buttyan, and S. Capkun. The Quest for Security in Mobile Ad Hoc Networks. In Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc) 2001.

[37] I. Ingemarsson, D.T. Tang and C. K. Wang. A Conference Key Distribution System. IEEE Transaction on Information Theory, 28(5): 107-125, 1992.

[38] P. Jacquet, P. Muhlethaler, A. Qayyum, A. Laouiti, L. Viennot, T. Clausen. Optimized Link State Routing Protocol (OLSR), RFC 3626. http://www.olsr.net/, http://www.olsr.org/

[39] N. Jiang, S. Sheu, K. A. Hua, O. Ozyer. A Finite-State-Model Scheme for Efficient Cooperation Enforcement in Mobile Ad Hoc Networks. The 11th IEEE International Conference on Parallel and Distributed Systems (ICPADS 2005).

[40] D. B. Johnson and D. A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. In T. Imielinski and H. F. Korth, editors, Mobile Computing, pages 153--181. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996.

[41] D.B. Johnson. Routing in Ad Hoc Networks of Mobile Hosts. In Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'94), 1994.

[42] G. T. Karetsos, S. A. Kyriazakos, and E. Groustiotis. A Hierarchical Radio Resource Management Framework for Integrating WLANs in Cellular Networking Environments. IEEE Wireless Communications Magazine. Vol 12. No. 6. December 2005.

[43] V. Kärpijoki, Security in Ad Hoc Networks. In Proceedings of the Helsinki University of Technology, Seminar on Network Security, 2000.

[44] Y.-B. Ko, V. N. H. Location-Aided Routing in mobile Ad hoc networks. In Proceedings of ACM/IEEE Mobicom, pages 66-75, October 1998.

[45] J. Kong, X. Hong, Y. Yi, J. S. Park, M. Gerla. A Secure Ad-hoc Routing Approach using Localized Self-healing Communities. In Proceedings of The Sixth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC 2005).

[46] Y. J. Lee and G. F. Riley. Dynamic NIx-Vector Routing for Mobile Ad Hoc Networks. Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC 2005), New Orleans, Mar. 13 - 17, 2005.

[47] B. Lehane, L. Doyle, and D. O'Mahony. Ad Hoc Key Management Infrastructure. International Conference on Information Technology: Coding and Computing (ITCC'05).

[48] A. Lindgren and O. Schelén. Infrastructured ad hoc networks In Proceedings of the 2002 International Conference on Parallel Processing Workshops (International Workshop on Ad Hoc Networking (IWAHN 2002)). pages 64-70. August 2002.

[49] U. Lu, B. Pooch. Cooperative Security Enforcement Routing in Mobile Ad Hoc Networks. In Mobile and Wireless Communications Network, 2002.

[50] E. M. Royer and C. Toh. A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks.

[51] B. S. Manoj, R. Ananthapadmanabha, and C. Siva Ram Murthy. Link life Based Routing Protocol for Ad hoc Wireless Networks. In Proceedings of The 10th IEEE International Conference on Computer Communications 2001 (IC3N 2001), October 2001.

[52] S. Marti, T.J. Giuli, K. Lai, and M. Baker. Mitigating Routing Misbehavior in Mobile Ad Hoc Networks. In Proceedings of MOBICOM 2000, pages 255-265, 2000.

[53] P. Michiardi and R. Molva. CORE: A Collaborative Reputation Mechanism to Enforce Node Cooperation in Mobile Ad Hoc Networks. Sixth IFIP Conference on Security Communications and Multimedia (CMS 2002).

[54] P. Michiardi and R. Molva. Prevention of Denial of Service Attacks and Selfishness in Mobile Ad Hoc Networks. Research Report N° RR-02-63. January 2002.

[55] P. Michiardi, R. Molva. Simulation-based Analysis of Security Exposures in Mobile Ad Hoc Networks. European Wireless Conference, 2002.

[56] P. Michiardi, R. Molva. A Game Theoretical Approach to Evaluate Cooperation Enforcement Mechanisms in Mobile Ad hoc Networks. WiOpt'03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks.

[57] S. Murthy, J. J. Garcia-luna-aveces. WRP (Wireless Routing Protocol) - A Routing Protocol for Packet Radio Networks, Proc. ACM International Conference on Mobile Computing and Networking, pp. 86-95, November, 1995.

[58] R. B. Myerson, Game Theory: Analysis of Conflict. Harvard University Press, Cambridge, Mass., 1991.

[59] R. K. Nichols and P. C. Lekkas. Wireless Security: Models, Threats, and Solutions. McGraw-Hill Inc, 2002.

[60] P. Papadimitratos and Z. J. Haas. Secure Routing for Mobile Ad Hoc Networks. In SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002), San Antonio, TX, January 2002.

[61] P. Papadimitratos and Z. J. Haas. Secure Data Transmission in Mobile Ad Hoc Networks. ACM Workshop on Wireless Security (WiSe) 2003.

[62] P.Papadimitratos and Z. Haas. Secure Link State Routing for Mobile Ad Hoc Networks. In Proceedings of the IEEE Workshop on security and Assurance in Ad Hoc Networks, in conjunction with the 2003 International Symposium on Applications and the Internet, 2003.

[63] J. Parker, J. L. Undercoffer, J. Pinkston, and A. Joshi. On Intrusion Detection in Mobile Ad Hoc Networks. In 23rd IEEE International Performance Computing and Communications Conference - Workshop on Information Assurance. IEEE, 2004.

[64] T. Parker and K. Langendoen. Guesswork: Robust Routing in an Uncertain World. In Proceedings of the 2nd IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS 2005), November 2005.

[65] V. Park, S. Corson. TEMPORALLY-ORDERED ROUTING ALGORITHM (TORA) VERSION 1 Internet Draft, draft-ietf-manet-tora-spec- 03.txt.

[66] C. E. Perkins, P. Bhagwat. DSDV (Highly Dynamic Destination-Sequenced Distance Vector routing protocol) - Highly Dynamic Destination-Sequenced Distance Vector (DTDV) for Mobile Computers Proc. of the SIGCOMM 1994 Conference on Communications Architectures, Protocols and Applications, Aug 1994, pp 234-244.

[67] C. Perkins, E. Royer and S. Das. AODV (Ad hoc On Demand Distance Vector routing protocol), RFC 3561.

[68] A. Perrig, R. Canetti, D. Song, and J. D. Tygar. Efficient and Secure Source Authentication for Multicast. In Network and Distributed System Security Symposium, NDSS '01, 2001.

[69] A. Perrig, R. Canetti, J.D. Tygar, and D. Song. Efficient Authentication and Signing of Multicast Streams over Lossy Channels. In IEEE Symposium on Security and Privacy, 2000.

[70] T. S. Rappaport. Wireless Communications Principles and Practice. Prentice Hall Inc. 1996.

[71] R. L. Rivest, A. Shamir, and L. M. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM, 21(2):120-126, 1978.

[72] N. Ben Salem, L. Buttyan, J. P. Hubaux, M. Jakobsson. A Charging and Rewarding Scheme for Packet Forwarding. In Proceedings of MOBIHOC 2003.

[73] C. Santivanez and R. Ramanathan. HSLS (Hazy Sighted Link State routing protocol) - Hazy Sighted Link State routing protocol (HSLS),BBN Technical Memorandum No. 1301, 31 August 2001. http://www.cuwireless.net/OSI/progress_report.html

[74] K. Sanzgiri, B. Dahill, Brian N. Levine, C. Shields, and E. M. Belding-Royer. A Secure Routing Protocol for Ad Hoc Networks. In Proceedings of 2002 IEEE International Conference on Network Protocols (ICNP). November 2002.

[75] R. S. Sisodia, B. S. Manoj, and C. S. R. Murthy. A Preferred Link Based Routing Protocol for Ad Hoc Wireless Networks. Journal of Communications and Networks, Vol. 4, No. 1, pp. 14-21, March 2002.

[76] V. Srinivasan, P. Nuggehalli, C. F. Chiasserini, R. R. Rao. Cooperation in Wireless Ad Hoc Networks. In Proceedings of 2003 of IEEE INFOCOM 2003.

[77] F. Stajano and R. Anderson. The Resurrecting Duckling. Lecture Notes in Computer Science. Springer-Verlag, 1999.

[78] M. Steiner, G. Tsudik, and M. Waidner, Key Agreement in Dynamic Peer Groups. IEEE Transactions on Parallel and Distributed Systems", Vol. 1, No. 8 (Aug 2000): pp 769-80.

[79] J. Z. Sun, J. Riekki, M. Jurmu, and J. Sauvola. Adaptive Connectivity Management Middleware for Heterogeneous Wireless Networks.

[80] A. S. Tanenbaum. Computer Networks 3rd Edition. Prentice Hall, Inc. 1996.

[81] G. Theodorakopoulos, J. S. Baras. Trust Evaluation in Ad-Hoc Networks. ACM Workshop on Wireless Security (WiSe) 2004.

[82] C-K Toh. ABR (Associativity-Based Routing protocol): A Novel Distributed Routing Protocol To Support Ad hoc Mobile Computing, Proc. IEEE 15th Annual International Phoenix Conference on Computers and Communications, IEEE IPCCC 1996, 27 March-29, Phoenix, AZ, USA.

[83] C-K Toh. Ad Hoc Mobile Wireless Networks Protocols and Systems. Prentice Hall PTR. 2002.

[84] C. Y. Tseng, P. Balasubramanyam, C. Ko, R. Limprasittiporn, J. Rowe, and K. Levitt. A Specification-based Intrusion Detection System for AODV. In Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks, 2003.

[85] Y.-C. Tseng, J. R. Jiang, and J H. Lee. Secure Bootstrapping and Routing in an IPV6-based Ad Hoc Network. In ICPP Workshop on Wireless Security and Privacy, 2003.

[86] A. Urpi, M. Bonucelli, and S. Giordano. Modelling cooperation in Mobile Ad Hoc Networks: A formal description of Selfishness. 1st International Symposium on Modeling and Optimization in Mobile Ad Hoc and Wireless Netowrks (WiOpt '03).

[87] A. C. Valera, W. K. Seah and S. V. Rao. CHAMP: A Highly-Resilient and Energy-Efficient Routing Protocol for Mobile Ad hoc Networks. In Proceedings of the 5th IEEE Conference on Mobile and Wireless Communications Networks (MWCN 2002), Stockholm, Sept 9 - 11, 2002.

[88] Joshua Wright, GCIH, CCNA. Detecting Wireless LAN MAC Address Spoofing.http://home.jwu.edu/jwright/papers/wlan-mac-spoof.pdf

[89] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed Energy Conservation for Ad Hoc Routing. In Proceedings of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (ACM Mobicom), July 16-21, 2001, Rome, Italy.

[90] Y. Xue, B. Li, and K. Nahrstedt. Price-based Resource Allocation in Wireless Ad Hoc Networks. 11th International Workshop on Quality of Service (IWQoS), 2003.

[91] H. Yang, X. Meng, Songwu Lu. Self-organized Network-layer Security in Mobile Ad Hoc Networks. ACM Workshop on Wireless Security (WiSe) 2002.

[92] Alec Yasinsac, Vikram Thakur, Stephen Carter, and Ilkay Cubukcu, A Family of Protocols for Group Key Generation in Ad Hoc Networks, Proceedings of the *IASTED International Conference on Communications and Computer Networks (CCN02)*, pp. 183-187, Nov.4-6, 2002.

[93] S. Yi, P.Naldurg, and R. Kravets. Security-aware Ad Hoc Routing for Wireless Networks. In Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking and Computing, ACM Press, 2001.

[94] M.G.Zapata and N. Asokan. Securing Ad Hoc Routing Protocols. ACM Workshop on Wireless Security (WiSe) 2002.

[95] X. Zeng, R. Bagrodia, and M. Gerla. GloMoSim: A library for parallel simulation of large-scale wireless networks. Proceedings of the 12th Workshop on Parallel and Distributed Simulations (PADS '98), May 26-29, in Banff, Alberta,Canada, 1998.

[96] Y. Zhang and W. Lee. Intrusion Detection in Wireless Ad Hoc Networks. In Proceedings of MOBICOM 2000, pages 275-283, 2000.

[97] W. Zhao, X. Li, and Y. Wang. Truthful multicast routing in selfish wireless networks. In Proceedings of MOBICOM 2004, pages: 245 – 259, 2004.

[98] W. Zhao, X. Li, and Y. Wang. Design Multicast Protocols for Non-Cooperative Networks. 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM05**).**

[99] L. Zhou and Z. Haas. Securing Ad Hoc Networks. In IEEE Network magazine, special issue on networking security, Vol. 13, No. 6, November/December, pages 24-30, 1999.

[100] S. Zhong, J. Chen, Y. R. Yang, Sprite: A Simple, Cheat-Proof, Credit-Based System for Mobile Ad-Hoc networks. IEEE INFOCOM 2003.

[101] ANSI/IEEE Standard 802.11, 1999 Edition. 1999. http://standards.ieee.org/catalog/olis/lanman.html.

[102] Information Technology Laboratory, National Institute of Standards and Technology. The Keyed-Hash Message Authentication Code (HMAC).

[103] I. S. Committee. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification. IEEE 802.11 Standard, IEEE, New York, ISBN 1-55937-935-9, 1997.

[104] Routing Information Protocol. RFC 1058. http://www.ietf.org/rfc/rfc1058.txt

[105] Routing Information Protocol version 2. RFC 1723. http://www.ietf.org/rfc/rfc1723.txt.