University of Strathclyde

Dealing with Free-Riding Nodes in an Open MANET

Khairul Azmi Abu Bakar

A thesis submitted for the degree of Doctor of Philosophy

Department of Electronic and Electrical Engineering

March 2012

Abstract

A mobile ad hoc network (MANET) is a self-organised wireless network where mobile nodes can communicate with each other without the need of any existing network infrastructure or centralised administration. There are no dedicated routing devices and each node has to rely on others to cooperatively provide forwarding services to ensure global connectivity. In an open MANET, different nodes may have different authorities and different goals. To conserve their own resources, some nodes may choose not to cooperate while still using the network to forward their packets. Uncooperative or misbehaving nodes can significantly degrade the performance of a MANET. Most previous works to mitigate the effects of misbehaving nodes focus on data forwarding. However, dropping control packets is a better strategy for the selfish nodes to avoid themselves from being requested to forward data packets. In this thesis, a new scheme is proposed to detect free-riding nodes which exhibit such a strategy. In the detection scheme, each node operates in promiscuous mode and monitors the activities of its neighbouring node. All nodes in the network are expected to contribute to the network on the continual basis within each predefined time frame. Those which fail will undergo a suspicious checking procedure to test for the suspicious behaviour. Detection alone is not enough to deal with free-riding nodes. To encourage the nodes to be cooperative, three variations of punishment schemes are proposed. The schemes punish free-riding nodes by isolating them and refusing them services.

Acknowledgements

First and foremost, I would like to thank to Almighty God for answering to my prayers and giving me the strength and will to complete this thesis. I would like to express my deepest gratitude and appreciation to my supervisor, Dr. James Irvine for his invaluable directions and support throughout my study. His insights and suggestions to the thesis topic have given me invaluable guidance in various stages of my work. His directions with regard to the developments of my academic thinking, problem solving and technical writing were inspiring and will be very helpful in my future work and study.

I am grateful for the support and assistance from all current and former members of the Communication Division especially the Mobile Group who have helped me in many ways and have given me a sense of belonging as a group.

I would also like to extend my sincere gratitude to my employer and sponsor MIMOS Berhad which is an agency under purview of Malaysian Ministry of Science, Technology and Innovation (MOSTI) for granting me the scholarship to pursue my PhD at University of Strathclyde.

Last but not least, my family deserves special recognitions for their unconditional support during the past years. I am grateful to my wife Asmaheram Che Awang, my daughter Nur Aisyah and my son Ahmad Luqman Hakim for their love, endless patience and encouragement in my graduate study. Without them, all that I have achieved would not have been possible.

List of Publications

The work reported in this thesis has produced the following conference publications:

1. K. A. A. Bakar and J. Irvine, "Contribution Time-based Selfish Nodes Detection Scheme", in *The 11th Annual Postgraduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting (PGNET2010)*, (Liverpool, UK), pp. 271–275, June 21-22 2010.

2. K. A. A. Bakar and J. Irvine, "A Scheme for Detecting Selfish Nodes in MANETs Using OMNET++", *in The 6th International Conference on Wireless and Mobile Communications (IEEE ICWMC 2010)*, (Valencia, Spain), pp. 410-414, September 20-25 2010.

3. K. A. A. Bakar and J. Irvine, "Coping with Free-Riding Nodes in Open MANET", submitted to the IEEE Transactions on Vehicular Technology.

Acronyms and Abbreviations

ACK	Acknowledgement
AODV	Ad hoc On-Demand Distance Vector
AP	Access Point
CBR	Constant Bit Rate
CCK	Complimentary Code Keying
DCF	Distributed Coordination Function
DoS	Denial-of-Service
DSDV	Destination-Sequenced Distance Vector
DSR	Dynamic Source Routing
DYMO	Dynamic MANET On-demand
FDR	False Detection Ratio
FHSS	Frequency-Hopping Spread Spectrum
GSM	Global Systems for Mobile Communications
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISM	Industrial, Scientific and Medical
LAN	Local Area Network
LR-WPANs	Low-Rate Wireless Personal Area Networks
MAC	Medium Access Protocol
MANET	Mobile Ad-hoc NETwork

NS-2	Network Simulator 2
OFDM	Orthogonal Frequency-division Multiplexing
OLSR	Optimized Link State Routing
OMNET++	Objective Modular Network Testbed in C++
OPNET	Optimized Network Engineering Tool
PAN	Personal Area Network
PDA	Personal Digital Assistant
RERR	Route Error
RF	Radio Frequency
RFC	Request For Comments
RO	Routing Overhead
RREP	Route Reply
RREQ	Route Request
TDPF	Total Data Packet Forwarded
TDR	True Detection Ratio
TETRA	Terrestrial Trunked Radio
TTL	Time to Live
UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunication System
UWB	Ultra Wide Broadband
WAN	Wide Area Network
WiMax	Worldwide Inter-operability for Microwave Access
WLAN	Wireless Routing Protocol

WPAN	Wireless Personal Area Network
WSN	Wireless Sensor Network
ZRP	Zone Routing Protocol

Contents

List of Figures

List of Tables

Chapter 1

Introduction

During the past decade, mobile computing and wireless communication technologies have been developing extremely fast due to the proliferation of inexpensive, widely available wireless devices. Current cellular systems have reached a high penetration rate, enabling worldwide mobile communication and Internet access. In addition, more and more wireless LAN hot spots are emerging, allowing people to surf the Internet in airports, railways, hotels and other public areas with their portable devices such as laptops and wifi enabled smartphones.

All these networks are conventional networks which depend on fixed network infrastructure and central administration. These networks require a large investment before they are operational and useful. Furthermore, updating these networks to meet continuously growing requirements such as bandwidth has proven to be quite expensive and slow. At the same time, more and more digital devices are produced which could be equipped with relatively short-range wireless transmission interfaces. These devices are becoming smaller, cheaper but more popular and powerful.

In the next generation of wireless communication systems, there will be a need for the rapid deployment of independent mobile users. Significant examples include establishing survivable, efficient, dynamic communication for emergency or rescue operations, disaster relief efforts and military networks. Such network scenarios cannot rely on centralised and organised connectivity and can be conceived as applications of wireless ad hoc network.

In order to enable multiple small portable devices to interconnect with each other without any fixed network infrastructure or central administration, a new alternative network architecture has been designed. In such architecture, the devices form a self-organizing and self-administering wireless network called a mobile ad hoc network [1].

1.1 Mobile Ad Hoc Networks

Mobile Ad-hoc Networks (MANETs) have attracted the attention of researchers due to their potential use in exciting new applications. A MANET is a collection of mobile devices that dynamically function as a network without the use of any existing infrastructure and centralised administration. It consists of wireless devices such as laptops, PDAs or mobile phones that come together to form a wireless network. Since the nodes are mobile, the network topology may change rapidly and unpredictably over time. There are no dedicated routers and all activities including discovering the topology and delivering messages must be executed by the nodes themselves. Nodes can communicate with their neighbours which are within wireless range as well as with distant nodes using intermediary nodes as routers [2]. Thus, nodes in such a network have to act as host or router or both at the same time.

There are a growing number of real time applications using ad hoc networks. Some of the potential applications of ad hoc networks are [3]:

- Conferencing: Mobile conferencing temporary network which could enable laptops or palmtop to spread or sharing information among participants in a conference

- Home Networking: It might be possible to deploy ad hoc technology to enable direct communication between devices at home. This would make possible the exchange of information such as voice, video-alarms and configuration updates

- Internet Hot Spots: Ad hoc networks can be linked to a fixed infrastructure via access points to provide extended wireless Internet access

- Personal Area Networks: Short-range ad hoc networks can be formed to simplify intercommunication between various mobile devices such as a mobile phone and a laptop by forming a personal area network (PAN)

- Emergency Services: Ad hoc networks can help to overcome network impairment during disaster emergencies. Mobile units will probably also carry traditional networking equipment in support of routine operations for the times when the Internet is available and the infrastructure has not been impaired.

- Vehicular Networks: Vehicles on a highway can form an ad hoc network in order to propagate information such as traffic and road conditions. This information can be generated by an individual vehicle and subsequently broadcast to other vehicles. Alternatively, the information can be transmitted to and received from fixed network access points placed near the road.

- Sensor Dust: Ad hoc networks can coordinate the activities and reports of a large collection of tiny sensor devices which could offer detailed information about terrain or environmental dangerous conditions. Networks of these sensors can be used in many different ways:

- Monitoring Space: Environmental and habitat monitoring, precision agriculture, indoor climate control, surveillance and intelligent alarms
- Monitoring Objects: Structural monitoring, condition-based equipment maintenance, media diagnostics, etc.
- Monitoring Interactions: Between objects and their environment, e.g. wildlife habitats, disaster management, emergency response, healthcare and manufacturing process flow.

1.2 Problem Statement and Motivation

Most current ad hoc routing protocols assume that the wireless network is benign and every node in the network strictly follows the routing behaviour and is willing to forward packets for other nodes. Most of these protocols cope well with the dynamically changing topology. These assumptions hold for applications such as military or search-and-rescue operations where all nodes belong to the same authority and share the same goals. However, application of MANETs for the support of open communities has emerged recently. In such an environment, different users with different goals are expected to cooperatively share the resources of their devices to ensure global connectivity.

However, in an open MANET, there is no guarantee that nodes in the network are willing to cooperate. Some resources such as battery power are scarce in the mobile environment and can be depleted with the device utilization. To extend the life of their devices, users exhibit selfish behaviour or misbehaviour by which they can benefit from the resources provided by the other nodes without making available the resources of their own devices in return.

Consider a campus wide wireless network with incomplete coverage. In order to preserve Internet access while moving within the campus, students are assumed to cooperate. If nodes accept to forward messages when located close to base stations, connectivity for the whole MANET may be ensured. However, if no mechanism is present to reward cooperation and discourage selfishness, some users may instruct their devices not to relay incoming messages, as this would extend their battery power and increases the available bandwidth. The cooperative nodes that are willing to forward messages for third parties obtain no benefit for their behaviour and realise that selfish nodes are implicitly rewarded. This motivates them to become selfish as well. When that happens, message forwarding may be disrupted to the point the MANET collapses.

In order to effectively support open and spontaneous communities, MANETs should complement routing protocols with additional mechanisms that reward co-operation and discourage misbehaving. In these open MANETs, each user will agree to share the resources of his node only if this brings him some benefits. While it is impossible to prevent selfish behaviour, it is possible to design mechanism that discourage such behaviour. This can only be achieved by applying some kind of penalty/reward to users.

The main objective of this work is to develop a scheme that allows each of the participating nodes to independently deal with free-riding nodes in an open MANET with minimal overhead. Free-riding nodes misbehave by dropping control packets from other nodes and as a result do not get any data forwarding requests. At the same time, free-riding nodes use forwarding services provided by cooperative nodes in the network to send their packets.

The first important step when dealing with free-riding nodes in a MANET is to identify such nodes from those that are cooperative or well-behaving. The work presented here proposes a new detection scheme to detect free-riding nodes in an open MANET. Each participating node independently monitors its neighbouring nodes through promiscuous mode to ensure they contribute to the network in every predefined time period. Those who fail to do so will be undergo a special testing called suspicious checking to give them a second chance before they will be labelled as free-riding nodes.

A detection scheme alone is not enough to discourage free-riding nodes from continuing with their misbehaviour. Free-riding nodes that have been detected should be punished for their misbehaving and prevented from using the network services. In this thesis, we proposed three variations of punishment schemes.

1.3 Contributions

The contributions of this thesis are summarized as the following:

1. The effects of misbehaving nodes to the performance of a MANET have been shown
2. A new scheme to detect free-riding nodes in a MANET is designed and analysed
3. Three variations of a punishment scheme to mitigate the effects of free-riding nodes are proposed and evaluated

1.4 Thesis Organisation

This thesis is divided into eight chapters. Chapter 2 begins with background information and classification of wireless communication. It presents more detailed descriptions about the advantages and security vulnerabilities of mobile ad hoc network compared to traditional fixed infrastructure networks. The classifications of routing protocols in MANETs, focussing on Dynamic Source Routing (DSR), the routing protocol used in this work, is also described in this chapter. Lastly, the descriptions of two types of misbehaving nodes (misleading and free-riding nodes) are presented.

In Chapter 3, the element of trust in a decentralised system such as mobile ad hoc network is introduced. Some mechanisms to detect and handle misbehaving nodes in mobile ad hoc networks that have been proposed in the literature are briefly described. The related works to detect misleading nodes are categorized into three groups: credit-based systems, reputation-based systems and acknowledgement-based systems. In addition, some published works designed to handle free-riding nodes are also presented.

In Chapter 4, OMNET++, the network simulator used in this work, is introduced. INETMANET framework which contains network model for DSR routing protocol is chosen to validate the behaviour of a MANET. The model is then modified to simulate the two types of misbehaving nodes. The effects of the misbehaving nodes to the performance of the MANETs in terms of packet delivery ratio (PDR) and total mean end-to-end delay are presented at the end of the chapter.

In Chapter 5, the proposed detection scheme to detect free-riding nodes in an open MANET is described. The assumptions used in the scheme are explained briefly. The monitoring nodes operate in promiscuous mode to monitor the actions of each of the neighbouring nodes. The actions are categorized into three groups: utilizing, contributing and neutral. In order not to be labelled as free-riding, the nodes have to contribute to the network in every predefined period of time. The performance evaluation of the proposed scheme is discussed in Chapter 6 to demonstrate the merit of the proposed scheme.

Free-riding nodes that have been detected need to be punished for their misbehaviour. In Chapter 7, three variations of punishment schemes intended for the detected free-riding nodes are proposed. The comparison between the punishment schemes is discussed in detail in this chapter. The punishments schemes are analysed using simulation models and their performance are compared with each other and with when no punishment scheme is used.

Finally, summary and conclusions of the thesis and some directions for future research are presented in Chapter 8.

Chapter 2

Background

2.1 Introduction

This chapter provides a brief review of background information and concepts about wireless networks (Section 2.2). The chapter also give more background review about MANETs (Section 2.3) and finally, network security issues particularly related to MANETs (Section 2.4).

2.2 Wireless Networks

In a simple words, a wireless network is a network in which devices communicate with each other without any wire, i.e. the communication medium is wireless. A device communicates with other devices by transmitting and receiving electromagnetic signals and the destination device must lie within the radio range of each other. The distance can be ranging from a few metres as in television remote control to millions of kilometres as for satellite communication. Other examples of wireless device

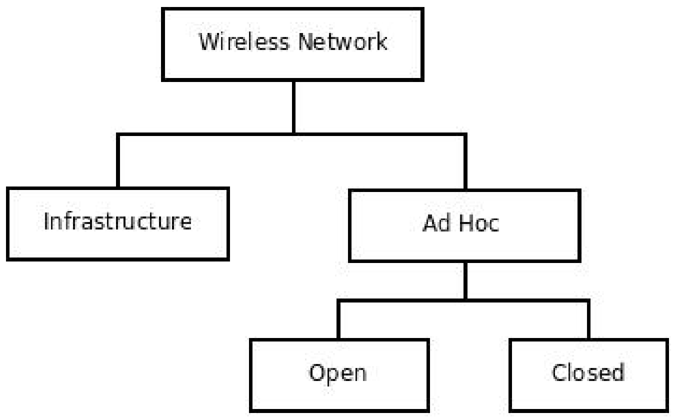


Figure 2.1: Classification of Wireless Networks

include mobile phones, wireless keyboards, satellite televisions, cordless telephones and wireless baby monitors.

Wireless networks allow mobility. Also, this feature allows the user to move freely while still enjoy the benefits of connecting to the network. Wireless signals can pass through walls which makes wireless networks an ideal solution in an area where cabling options are strictly limited. Installing wireless networks can be fast and easy as it can eliminate the need to pull cables through walls and ceiling. It is easier and more cost-efficient to install and operate compared to a corresponding wired network.

2.2.1 Classification of Wireless Network

Wireless networks can be classified into two categories: infrastructure based and ad hoc. In an infrastructure based network, the whole network is managed through dedicated supporting equipment such as base stations or access points. All communications are first sent to this centralised station before they are forwarded to the intended destination. Wireless nodes heavily rely on the centralised station and have

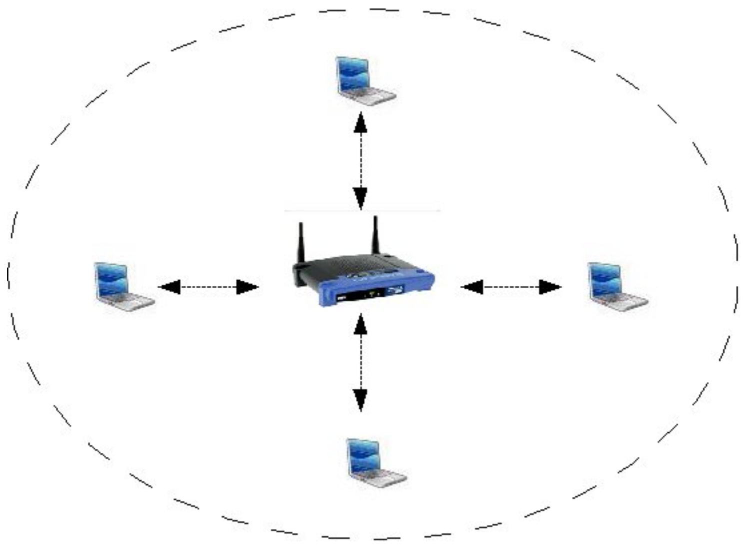


Figure 2.2: Infrastructure Based Wireless Network

to be within the base communication radius to be connected. If the base station is down or malfunctions, no node can communicate. Typical examples of this kind of wireless networks are GSM [4], UMTS [5] and TETRA [6].

An infrastructureless based network or ad hoc network on the other hand does not depend on pre-existing infrastructure or centralised base station to administrate and maintain the network. Without an inherent infrastructure, all network operations such as discovery the routes and delivering packets have to be done by the nodes themselves, either individually or collectively. In a pure ad hoc network, the nodes send data to one another directly. In this single-hop communication, the sender and the receiver have to be in the same transmission radius. If any one of them moves out of this radius, the communication will be terminated. In a multi-hop ad hoc network, the communication is extended as the intermediate nodes are expected to help forwarding packets to the next hop in order to deliver data across the network. As a result, a sender could send data to a receiver that located far beyond its transmission radius with the cooperation of one or more intermediate nodes. The nodes in ad hoc networks can be mobile phones, PDAs and laptops and typically support

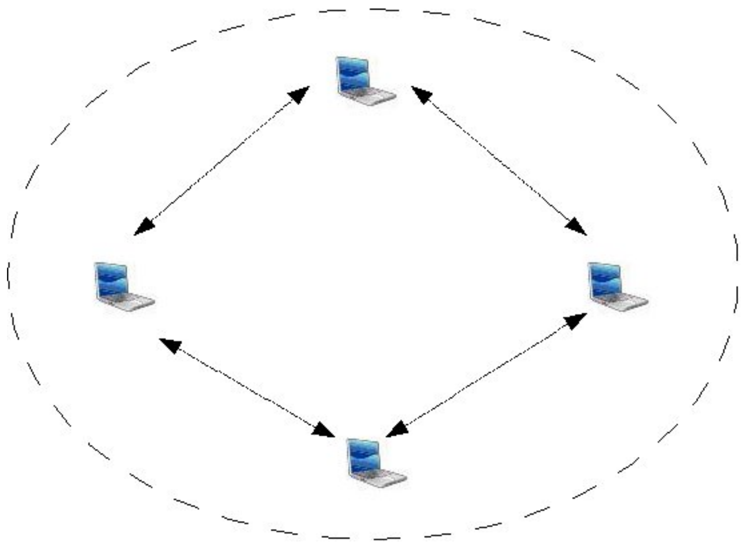


Figure 2.3: Infrastructureless Based Wireless Network

several forms of wireless connectivity.

Ad hoc networks are more suitable than infrastructure based networks in certain environments. Ad hoc networks are advantageous in situations where there is no network infrastructure available and there is a urgent need for users to communicate using mobile devices. For example, this could be in a scenario where a natural disaster like an earthquake devastates an area and the existing network infrastructures are destroyed. The rescue team would require a communication network to be deployed almost instantly to assist them in their operation. Another example is where people find themselves in a place like a conference room and need temporary network to communicate between their mobile devices. An ad hoc network would be more practical as it can be easily set up and broken down when it is not needed anymore.

2.3 MANETs

In this section, we will discuss the wireless communication technologies used in MANETs and more background information about MANETs particularly the classifications, features and advantages of MANETs. Then we will present issues with routing related to MANETs along with an example of a routing protocol.

2.3.1 MANET Technologies

Typical wireless communication technologies used in MANETs are:

- Wi-Fi (IEEE 802.11) [7]

  Wi-Fi uses 802.11 networking standards which come in several flavours.

  1. IEEE 802.11b: This is the first version to reach the marketplace. It is the slowest and least expensive standard. It is now becoming less common as faster standards become less expensive. IEEE 802.11b transmits in the 2.4GHz frequency band. It can handle data up to 11 Mbps and uses complimentary code keying (CCK) coding.
  2. IEEE 802.11g: It also transmits at 2.4 GHz frequency band but faster than IEEE 802.11b. It can handle data up to 54 Mbps by using a more efficient coding technique called orthogonal frequency-division multiplexing (OFDM). IEEE 802.11g is backward compatible and can handle IEEE 802.11b clients.

3. IEEE 802.11a: It transmits at 5 GHz frequency band and can handle the same data rate as IEEE 802.11g which is at 54 Mbps. It also use OFDM coding. However, it is not compatible with IEEE 802.11b.
4. IEEE 802.11n: The newest 802.11 standard with a significant increase in the maximum raw data rate to 600 Mbps. It uses OFDM modulation and operates in the 2.4 GHz band.

- Bluetooth (IEEE 802.15.1) [8]

  Bluetooth is a standard for short range, low power, low cost wireless technology. A lot of devices such as mobile phones, laptops, printers and digital cameras use Bluetooth as a cheap and easy way to connect and exchange information between themselves. A Bluetooth network is called piconet, consisting of one master and up to seven active slaves at one time. Bluetooth piconets can be extended by joining other piconets to form a multi-hop ad hoc network called a scatternet. Bluetooth uses the Frequency-Hopping Spread Spectrum (FHSS) technique and operates within the 2.4GHz band. Its capability includes a data transmission rate of 1 Mbps.

- ZigBee (IEEE 802.15.4) [9]

  ZigBee technology is a low cost, low power consumption and short distance wireless communication technology based on the IEEE 802.15.4-2003 standard for Low-Rate Wireless Personal Area Networks (LR-WPANs). The technology defined by the ZigBee specification is intended to be simpler and less expensive than other WPANs, such as Bluetooth. The technology allows for devices

Table 2.1: MANETs technologies comparison

Technology	Frequency Band	Theoretical maximum throughput	Average geographic range
Wi-Fi (IEEE 802.11b)	2.4 GHz	11 Mbps	100 metres
Wi-Fi (IEEE 802.11a)	5 GHz	54 Mbps	20 metres
Wi-Fi (IEEE 802.11g)	2.4 GHz	54 Mbps	100 metres
Wi-Fi (IEEE 802.11n)	2.4 GHz or 5 GHz	65 - 600 Mbps	up to 400 metres
Bluetooth (IEEE 802.15.1)	2.4 GHz	1 Mbps	10 metres
Zigbee (IEEE 802.15.4)	2.4 GHz, 915 MHz, 868 MHz	0.25 Mbps	up to 10 metres

to communicate with one another with very low power consumption, allowing the devices to run on simple batteries for several years. ZigBee operates in the 2.4 GHz, 915 MHz and 868 MHz ISM radio bands with data rate is up to 250 kbps. Its focus is to define a general-purpose, inexpensive, self-organizing, mesh network that can be used for industrial control, embedded sensing, medical data collection, smoke and intruder warning, building automation, home automation, and domotics, etc [10].

A comparison of different MANETs technologies according to their frequency bands, theoretical maximum throughput and communicate range are presented in Table 2.1.

2.3.2 Classification of MANETs

Ad hoc network can be further classified into two categories: closed and open [11]. In a closed ad hoc network, all nodes belongs to a single authority or company.

The nodes cooperate with each other towards a common goal such as in emergency search/rescue or military and law enforcement operations. In an open ad hoc network, the network is open for any participant that may appear to be a stranger to each other. In such environment, different nodes without predefined trust relationships are expected to share the responsibility of network formation and management.

One example of a closed ad hoc network is a wireless sensor network (WSN)[12]. A WSN is a self-configuring network of small sensor nodes communicating cooperatively amongst themselves using radio signals to monitor physical or environment conditions. The sensors used in WSNs are normally small, low-cost embedded devices equipped with an antenna, radio frequency (RF) transceiver to allow communication with other nodes, a memory unit, a CPU, the sensor unit and the power source which is usually a battery. Different sensor nodes can capture data such as temperature, humidity, motion, vibration or pressure. These sensors are deployed in a large number and can be scattered in a geographical region.

The WSN communication model involves routing to a sink [13]. Sensors in a WSN collect information from their environment and relay the data between themselves to a specialized node referred to as the sink node. The sink node is a sensor node with gateway functions to link to external networks such as the Internet. The applications for WSNs are varied, typically involving some kind of monitoring, tracking or controlling.

A mobile ad hoc network (MANET) is one typical open ad hoc network. A mobile ad hoc network is a self-configuring network of mobile nodes communicating amongst themselves through a wireless medium. A mobile ad hoc network is the mobile version of ad hoc network. Nodes in a MANET are free to move around in an arbitrary fashion resulting in the topology of the network changing dynamically and unpredictably. In an open MANET, the network is formed by any mobile devices that are located close enough. These mobile nodes may belong to a different authority or company and may have different goals when connected to the network.

2.3.3 Mobile Ad Hoc Network Features

MANET has several characteristics that differentiate it from other type of networks. Some of features of a MANET are as follows:

1. **Autonomous terminal**: MANET does not depend on any established infrastructure or centralised administration. Each node operates in distributed mode, acts as an independent router and generates independent data. In other words, besides the basic processing ability as a terminal, the mobile nodes can also perform switching functions just like a router. This means that endpoints and switches are usually indistinguishable in a MANET.

2. **Distributed operations**: Network management has to be distributed across different nodes, which brings added difficulty in fault detection and management.

3. **Multi-hop routing**: No default router is available, so every node acts as a router and forwards each others' packets to enable information sharing between mobile hosts.

4. **Dynamically changing network topologies**: In mobile ad hoc networks, because nodes can move arbitrarily, the network topology, which is typically multi-hop, can change frequently and unpredictably, resulting in route changes, frequent network partitions and possible packet losses.

5. **Variation in link and node capabilities**: Each node may be equipped with one or more radio interfaces that have varying transmission/receiving capabil-

ities and operate across different frequency bands [14, 15]. This heterogeneity in node radio capabilities can results in possibly asymmetric links. In addition, each mobile node might have a different software/hardware configuration, resulting in variability in processing capabilities. Designing network protocols and algorithms for this heterogeneous network can be complex, requiring dynamic adaptation to the changing conditions (power and channel conditions, traffic load/distribution variations, congestion, etc.)

6. **Energy constrained operation**: Because the batteries carried by each mobile node have limited power, processing power is limited, which in turn limits services and applications that can be supported by each node. This becomes a bigger issue in mobile ad hoc networks because, as each node is acting as both and end system and a router at the same time, additional energy is required to forward packets from other nodes.

7. **Network scalability**: Currently, popular network management algorithms were mostly designed to work on fixed or relatively small wireless networks. Many mobile ad hoc applications involve large networks with tens of thousands of nodes, as found for example, in sensor networks and tactical networks [16]. Scalability is critical to be successful deployment of these networks. The steps toward a large network consisting of nodes with limited resources are not straightforward and present many challenges that are still to be solved in areas such as addressing, routing, location management, configuration management, interoperability, security, high-capacity wireless technologies, etc.

2.3.4 Advantages of Mobile Ad Hoc Network

Mobile ad hoc networks have received a lot of attention from researchers in recent years. They are attractive due to the fact that they have certain interesting advantages over the traditional infrastructure network. Some of these advantages are:

- Use of ad-hoc networks can increase mobility and flexibility, as ad-hoc networks can be easily installed and removed when they are not needed anymore in a very short time.
- Ad-hoc networks can be more economical in some cases as they eliminate the cost of installation and maintenance of fixed network infrastructure.
- Ad-hoc networks can be more robust than conventional wireless networks because of their distributed nature, node redundancy and the lack of single point of failure.
- Because of multi-hop support in ad-hoc networks, communication beyond the Line of Sight (LOS) is possible at high frequencies.
- Multi-hop ad-hoc networks can reduce the power consumption of wireless devices. More transmission power is required for sending a signal over any distance in one long hop than in multiple shorter hops.
- Utilizing short communication links with multi-hop node-to-node communication instead of long-distance node to central base station communication, radio emission levels can be kept low. This reduces interference levels, increases spectrum reuse efficiency and makes it possible to use unlicensed unregulated frequency bands.

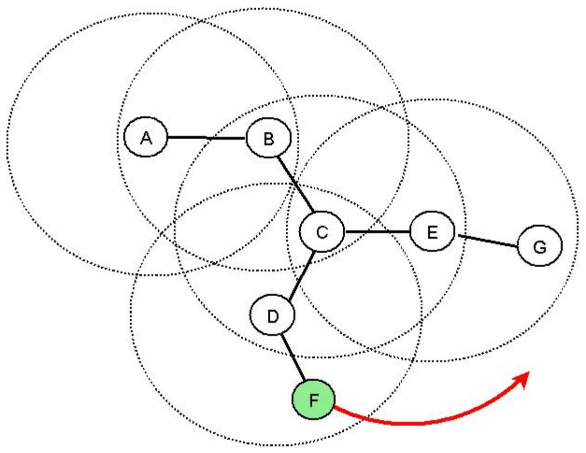
2.3.5 Routing in Mobile Ad Hoc Network

The routing protocol in an ad hoc network is a convention that controls how nodes come to agree which way to route packets between two nodes [17]. Packets are sent via the communication channels from source to the destination. That connection between source and destination is called a route or path. The route is composed of at least two nodes; the source which is the node that initiates the communication and the destination which is the target to receive the communication.

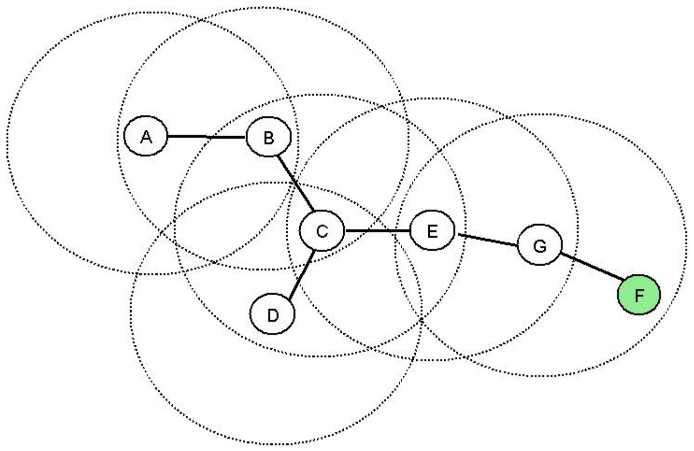
In wireless communication, if the source and the destination are within the transmission range, direct communication can take place. However, if the source and the destination are not in close proximity to each other to allow direct communication, multi-hop communication is required where intermediate nodes have to play their part in relaying packet. In this scenario a route will be composed of more than two nodes. The methods that nodes use to connect each other and to forward packets for each other are handled by routing protocols.

The absence of any central coordinator or base station makes routing in an hoc network a huge challenge compared to any infrastructure networks [18]. In an ad hoc network, there is no special router and each ad hoc node is expected to function both as a network host for transmitting and receiving data and as a network router for routing packets for other nodes. Routing protocols in an ad hoc network specifies how nodes in a network disseminate information with each other and enable them to make adjustments to the route selection based on their conditions.

The major challenges in routing for mobile ad hoc networks are imposed by the resources constraint and the mobility of the nodes participating in the network. Due to the mobility of the nodes, the connectivity between any two nodes is always considered temporal. When the two nodes move away from each other beyond the



(a)



(b)

Figure 2.4: Effect of Mobility in MANET

transmission radius, the link will be broken and alternative valid route should be used to ensure the connection.

Figure 2.4 illustrates the change of the network topology due to the node mobility. The circle represents the radio range of each node. Node F (represented by a green round ball) was initially connected to the network through node D as shown in Figure 2.4a. However, as it moves away to the right, it would eventually move out of the radio range of node D. Node F is then connected through node G. Now the new network topology has changed as illustrated in Figure 2.4b.

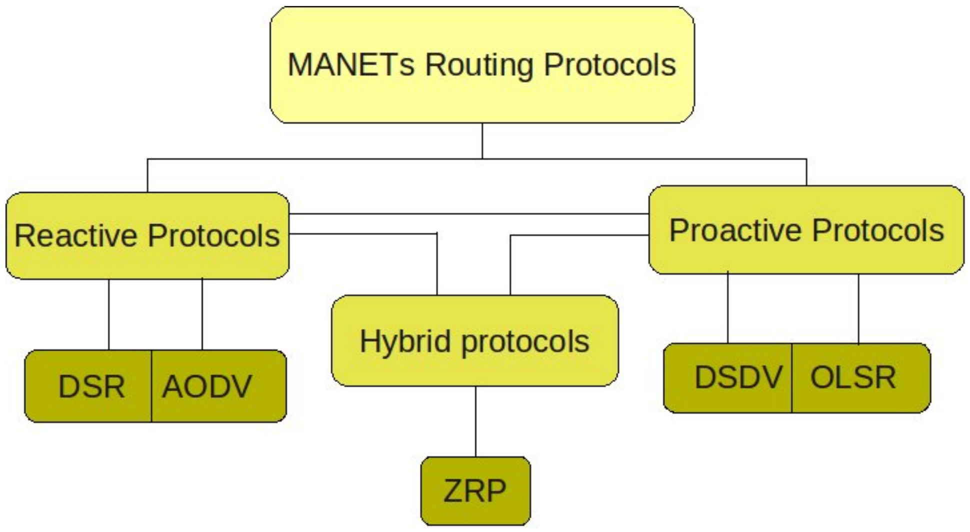


Figure 2.5: MANETs Routing Protocols

2.3.6 Classification of MANETs Routing Protocols

Routing protocols in MANETs can be categorized into three different categories according to their route creation and maintenance.

1. Proactive protocols
2. Reactive protocols
3. Hybrid protocols

The hierarchy of these protocols is shown in Figure 2.5.

In the following section, we will review each category of the routing protocols, how they work and the differences between them.

2.3.6.1 Proactive Routing Protocol

Proactive routing protocols are also referred to as table-driven protocols because each node creates a routing table for its neighbours and other nodes within the network. In proactive routing protocols, information about the current network topology is distributed periodically throughout the network. Each node then uses this information to compute and maintain routes to various destinations in the network. The aim of these protocols is for each node to have a valid route to each destination at all times even though some of the routes may not be used. One advantage of proactive routing protocols is that a node would experience minimal delay whenever a valid route is needed as it can be immediately obtained from the routing table. On the other hand, periodic updates are needed for distributing network topology information and this costs bandwidth and battery life of the nodes. Examples of routing protocols in this category are the Ad-hoc Wireless Distributing Service (AWDS) protocol [19] and the Destination Sequenced Distance Vector (SDV) protocol [20]. Proactive routing protocols are most suitable in ad hoc network where number of nodes is small and the nodes have limited mobility since routing update updates are not then sent very frequently.

2.3.6.2 Reactive Routing Protocol

Reactive routing protocols are also known as on-demand routing protocols. With a reactive protocol, a source node sends a request message for routing paths to the destination only when such path is required. Reactive routing protocols are divided into a route discovery cycle and route maintenance procedures. The route discovery cycle typically involves flooding some form of route request message through the network. Intermediate nodes or the destination node itself will send a reply message providing the initiator the route to the destination. Route maintenance is used to detect link breakage since there are no periodic route update messages unlike in proactive routing protocols. When a link break is detected between two nodes, one or both respective nodes are responsible for sending an information message about the broken link to all affected parties. A key advantage of reactive routing protocols is that they consume less bandwidth and battery power as compared to proactive routing protocols. However, reactive routing protocols have a certain amount of transmission delay since the packets have to wait while the node tries to find the route to the destination. Examples of reactive routing protocols are Dynamic Source Routing (DSR) protocol [21] and Ad-hoc on Demand Distance Vector (AODV) routing protocol [22]. Reactive routing protocols are most suitable for an ad hoc network where the network topology is dynamic and the number of nodes is large.

2.3.6.3 Hybrid Routing Protocol

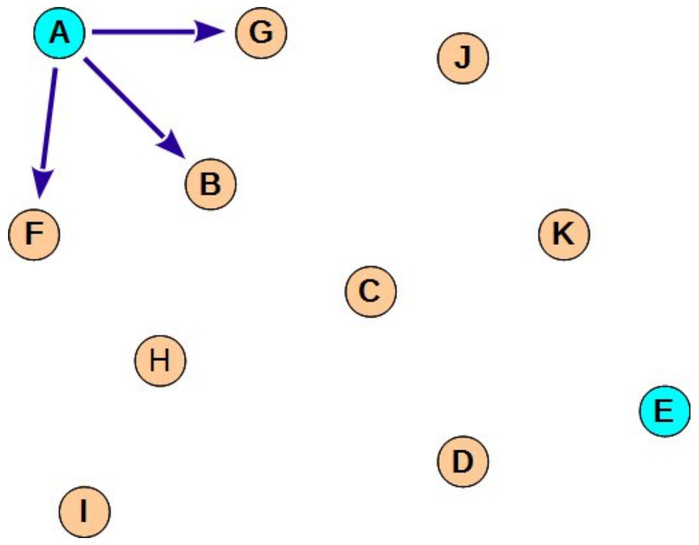
There are also hybrid routing protocols that basically combine the advantages of both proactive and reactive routing protocols. One popular example routing protocol under this category is called Zone Routing Protocol (ZRP) [23] which works by dividing the network into zones. Routing between different zones uses reactive routing protocols to cope with frequent node mobility. Route discovery and route maintenance procedure is required for destinations that are in other zones. On the other hand, routing within the same zone uses a proactive approach since it is assumed that the nodes in the same zone are relatively static. As a result, a route to a destination that is in the same zone could be established without delay. Routing protocols in this category can provide a better trade-off between communication overhead and delay but this trade-off is subject to the size and the dynamics of a zone. Thus, the hybrid approach is an appropriate candidate for routing in a large network.

2.3.7 Dynamic Source Routing (DSR)

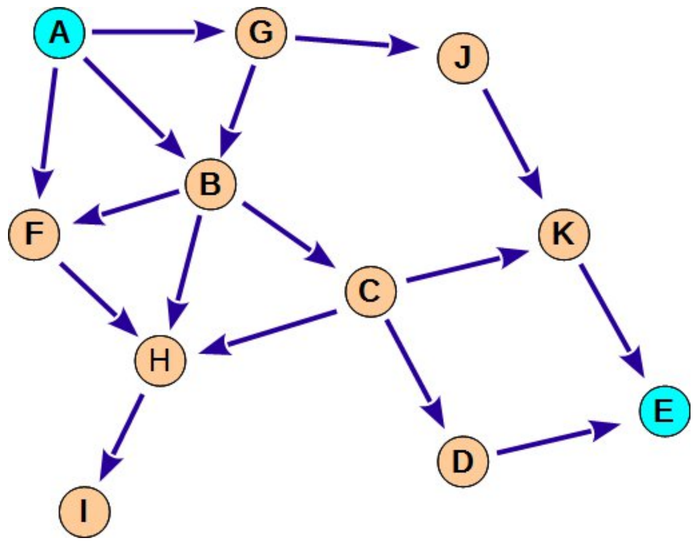
DSR is a simple but efficient reactive routing protocol specifically designed for use in multi hop wireless ad hoc networks of mobile nodes. It is an on-demand protocol because route paths are discovered at the time a source sends a packet to a destination for which the source has no path. Using DSR, the network becomes self-organizing and self-configuring, without the need to have a centralised administration. DSR is considered to be one of the best performing routing protocols in mobile ad hoc networks [24]. DSR is designed to work well even with high rates of mobility. It can also support up to one hundred nodes which means it can work well over medium network density [25].

The protocol is divided into two main phases: Route Discovery and Route Maintenance. Figure 2.6 illustrates an example of Route Discovery in which a node A wishes to form communication with node E but has no information about any path to node E. In this case, we call node A the initiator and node E the target of the Route Discovery. Node A initiates Route Discovery by transmitting a ROUTE REQUEST (RREQ) message as a single local broadcast packet which is received by (approximately) all nodes currently within wireless transmission range of node A. Each ROUTE REQUEST message has information about the initiator and target of the Route Discovery and also contains a unique ID generated by initiator. In addition to that, each ROUTE REQUEST message also contains a record listing the address of each intermediate nodes through which this particular copy of the ROUTE REQUEST has been forwarded. This route record is initialized to an empty list by the initiator of the Route Discovery.

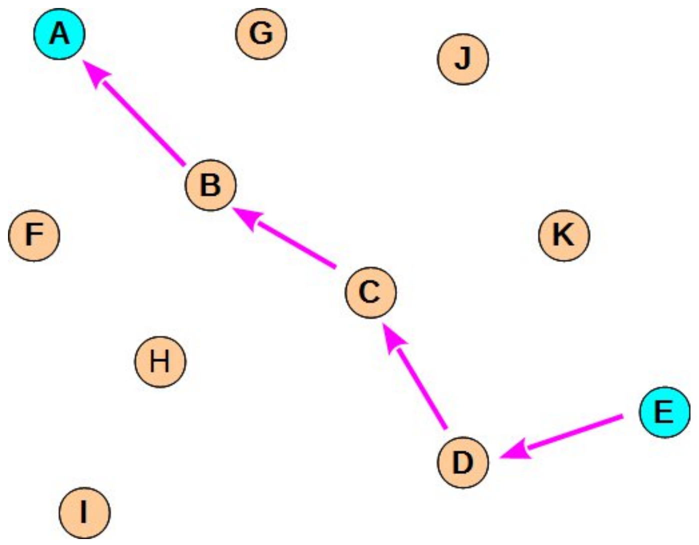
Each node, upon receiving the ROUTE REQUEST packet, appends its own addresses to the ROUTE REQUEST packet and rebroadcast the packet to its neigh-



(a)



(b)



(c)

Figure 2.6: Route Discovery Phase in DSR

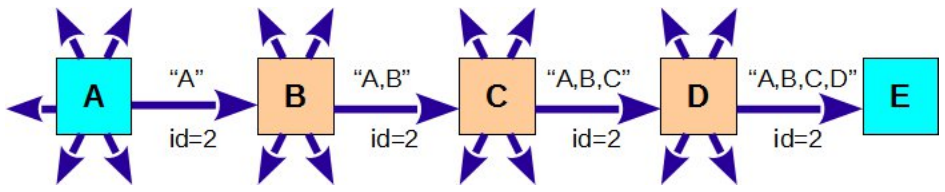


Figure 2.7: Simplified Route Discovery Example

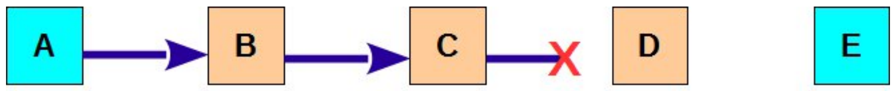


Figure 2.8: Route Maintenance Example

bours provided that its address is not in the record listing already and it is not target of the ROUTE REQUEST. This is necessary to prevent loop formation and to avoid multiple transmission of the same ROUTE REQUEST by an intermediate node. The process continues until the packet reaches the target node (E). Node E now must send back a ROUTE REPLY packet to inform node A about the discovered route. Normally, this is done using the route obtained by reversing the route appended in received ROUTE REQUEST packet. At this stage, the source node already has the route to the destination and can now start sending the data packet.

Since there could be more than one possible route from the source to the destination, node E may receive multiple route replies from node A. Normally, the source node will select the route from the first ROUTE REPLY received. The route information in the ROUTE REQUEST and ROUTE REPLY is used by every node to learn about routes to other nodes in the network.

The route maintenance function handles link breaks. When originating or forwarding a packet using the source route, each node transmitting the packet is responsible for confirming that the packet has been received by the next hop along the source route. For instance, in the scenario illustrated in Figure 2.8, node A has originated a packet for E using a source route through intermediate nodes B, C and D. In this case, node A is responsible for receipt of the packet at B, node B is responsible for receipt at C, node C is responsible for receipt at D, and node D is responsible for receipt finally at the destination E. The transmitting node will store a copy of the packet in a maintenance buffer. The packet will remain in the buffer until the responsible node receives the confirmation of receipt or after the packet has remain in the buffer longer than a certain period. This confirmation of receipt may be provided by either the following three ways as specified in the DSR draft [26]:

1. Link-layer acknowledgement: this is supplied by the MAC layer.
2. Passive acknowledgement: this confirmation comes indirectly by overhearing the next node forward the packet
3. Network-layer acknowledgement: this is when nodes explicitly request a DSR acknowledgement from the next hop

After a certain number of attempts to send the packet without any acknowledgement, the transmitting node will consider the link broken. If that happen, the responsible node sends a ROUTE ERROR (RERR) packet to the source node of the message. The notified source node then removes from its route cache all paths that use the broken link. It must also try another route or execute route discovery process again if it does not have any other routes. Nodes that get hold of the ROUTE ERROR packet would then update their routing tables accordingly.

2.4 Network Security: Overview

Security in a computer system is about the ability of the system to manage, protect and distribute sensitive information. The communication medium is normally shared with others and it is important that this untrusted channel be used in such a way as to ensure secured communication can take place.

This section discusses about technical security goals and possible attacks in computer network. In general there are six technical security goals [27].

1. Authentication: The identity of both the sender and the receiver must have been verified before the data is sent, i.e., that each is actually the party that it claims to be. The service must assure that the connection is not interfered with in such a way that a third party can masquerade as one of the legitimate parties for the purpose of unauthorized transmission or reception.

2. Confidentiality: The content of a message should only be disclosed to the authorized party. It should be protected from a passive attack where an attacker eavesdropping the communication is able to read the message.

3. Integrity: There should be an assurance that the content of the data is free from any type of modification between the end points. Modification includes writing, changing, deleting, creating and replaying the transmitted messages.

4. Non-Repudiation: It must be possible to identify the entity that is responsible for a particular event. This prevents either sender or receiver from denying sending or receiving a transmitted message.

5. Authorization: Only the authorized entities should be able to access certain services and data. Before an entity is allowed to gain access, it must first be

identified or authenticated so that the system can know the identity of the requester and assign the corresponding access right.

6. Availability: The services in the system should be available and function properly with quality of service expected despite of various attacks.

Attacks on a computer system are a violation of one or more of these security goals.

2.4.1 Security in MANETs

Securing MANETs is a challenging research problem. MANETs are prone to the same type of security vulnerabilities as the traditional wired networks. In addition, MANETs also face other security threats due to the key characteristics that MANETs have. From a security perspective, some of the key characteristics that differentiate MANETs from wired networks are:

- Open peer-to-peer network architecture,
- highly dynamic network topology,
- no clear line of defence,
- shared wireless medium, and
- stringent resource constraints

One fundamental vulnerability of MANETs comes from their open peer-to-peer architecture. Unlike traditional wired networks, MANETs do not have dedicated routers where traffic monitoring and access control mechanism can be deployed. In addition, in MANETs, each node in a MANETs may operate as a router and participate in routing and packet forwarding. The topology of MANETs is also more dynamic than wired networks. Nodes in MANETs are allowed to move around in an arbitrary fashion and even have freedom to join or leave the network at any time. Although MANETs can be more robust because of these unique characteristics, securing the network will be more challenging. There is no well defined place or infrastructure to deploy a single security solution and a malicious node along the route can tamper with routing and data packets.

The wireless channel that is used for the communication medium exposes the network to additional new risks compared to traditional wired networks. Unlike traditional wired networks where the physical transmission medium can be secured, wireless networks propagate the signal through the air. While this gives the users the freedom to move around while still connected to the network, it also allows easy access to potential attackers. It is hard to contain the signal from going out from its desired area. To attack a network of a company for example, the attacker does not have to be physically be inside the building. The attacker could launch the attacks such as eavesdropping and radio jamming even from the parking lot of the targeted building.

The stringent resource constraints in MANETs constitute another nontrivial challenge from a security perspective. The limitation of resources includes the wireless channel which is bandwidth-constrained and shared among multiple networking entities. Some nodes in MANETs may also have limited computation capability. Low-end devices such as PDAs can hardly perform computation-intensive tasks especially involving cryptographic computation. Most of the mobile nodes in MANETs typically operate using battery power and have very limited energy resources. Any

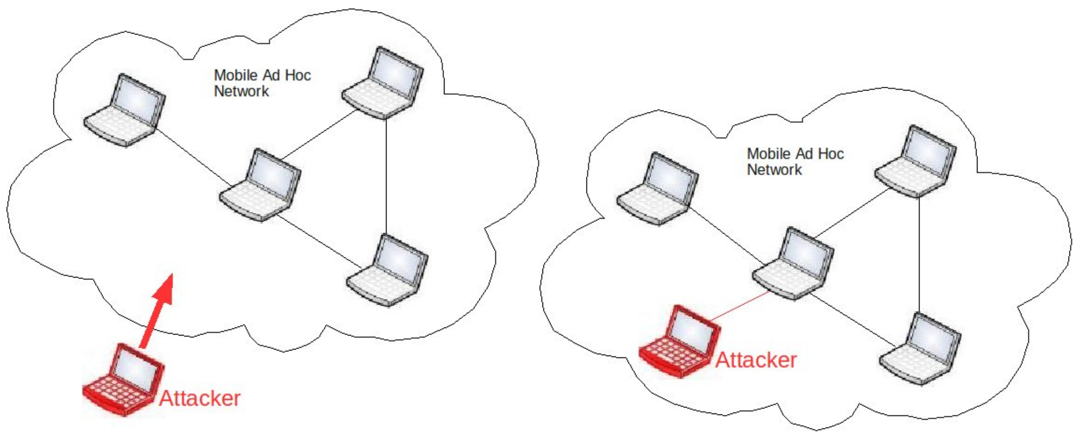


Figure 2.9: External and Internal Attacks in MANETs

security mechanisms will have to be able to operate at a reasonable speed using the limited computation capability and energy resources that may exist.

2.4.2 Classifications of Attacks

There are numerous kinds of attacks in MANETs. These security attacks can be grouped based on different characteristics. One way to differentiate attacks is to classify them by their sources as illustrated in Figure 2.9: External attacks and internal attacks.

- External attacks are carried out by parties that are not legally part of the network. The targeted network might be a self-contained entity, that is linked to other networks using the same infrastructure or communication technology. This would make it possible to initiate attacks without even being authenticated in the targeted network. On the other hand, it would also be possible to jam the communication of the entire network of a company from the parking lot in front of the company building.

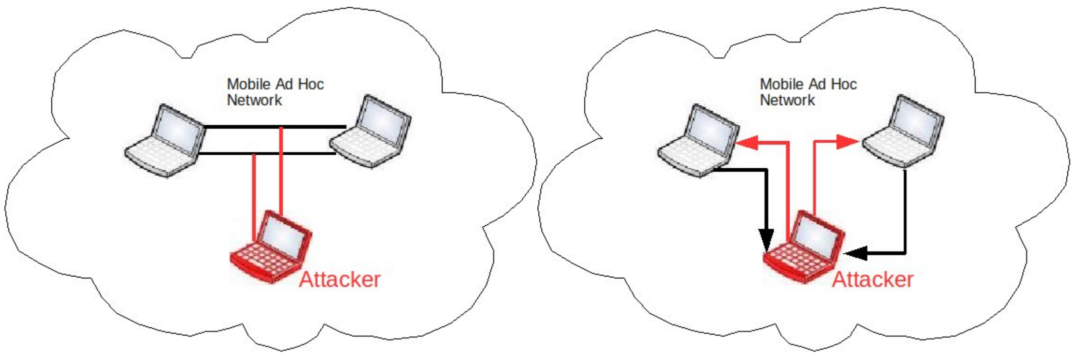


Figure 2.10: Passive and Active Attacks in MANETs

- Internal attacks on the other hand are carried out by compromised nodes which are actually part of the network. Internal attacks are more severe and not as easy to prevent when compared with external attacks since the attacker has the necessary information to participate in distributed operations and possesses privileged access rights.

Attacks in MANETs can also be classified into two major categories, namely passive attacks and active attacks, according to the attack means [28, 29]. The distinction between passive and active attacks is depicted in Figure 2.10.

- Passive attacks do not involve any disruption of the services. They only intended to gather data exchange and to eavesdrop on the communication within the network without disrupting the operation of the network. Passive attacks do not incur any additional energy cost. Examples of passive attacks are eavesdropping, traffic analysis and traffic monitoring.
- Active attacks on the other hand actively alter the data with the intent of overloading the network, obstructing the operation or to cut off certain nodes from their neighbours so they can not use the network services effectively any-

Table 2.2: Security Issues for Each Protocol Stack Layer

Layer	Security Issues
Application layer	Detecting and preventing viruses, worms, malicious codes, and application abuses
Transport layer	Authenticating and securing end-to-end communications through data encryption
Network layer	Protecting the ad hoc routing and forwarding protocols
Link layer	Protecting the wireless MAC protocol and providing link-layer security support
Physical layer	Preventing signal jamming denial-of-service attacks

more. To execute active attacks, the attacker must bear some energy cost to inject packets into the network. Examples of active attacks are jamming, impersonating, modification and denial of service (DoS).

Attacks can also be categorized based on network protocol layer. Table 2.2 describes the security issues in each layer as in [30]. In order to achieve the security goals, the security solution should provide complete protection spanning the entire protocol stack. However, in this chapter, we focus only on the network layer security issues.

2.4.3 Attacks in the Network Layer

Attacks in the network layer are considered as internal attacks because the attackers are part of the network. In the network layer, the main operations are ad hoc routing and data packet forwarding. Both of these operations interact with each other to fulfil the functionality of delivery of packets from source to the destination. Accordingly, attacks on the network layer, in general, fall into one of two categories: routing attacks and packet forwarding attacks.

Routing attacks generally involve the malicious manipulation of route advertisements and injection of false or incomplete information. The specific attack behaviour is related to the routing protocols used by the MANETs. For example, in the context of DSR, the attacker may modify the source route listed in the RREQ or RREP packets. It can delete a node from the list, switch the order, or append a new node into the list. In AODV, the attacker may advertise a route with a smaller distance metric than the actual distance, or advertise a routing update with a large sequence number and invalidate all routing updates from other nodes.

By attacking the routing protocols, the attackers can attract traffic toward certain destinations in the nodes under their control, and cause the packets to be forwarded along a route that is not optimal or even nonexistent. The attackers can create routing loops in the network, and introduce severe network congestion and channel contention in certain areas. Multiple colluding attackers may even prevent a source node from finding any route to the destination, and partition the network in the worst case.

More sophisticated and subtle routing attacks have been identified in recent research. For example, the attacker may further subvert existing nodes in the network, or fabricate its identity and impersonate another legitimate node [31]. A pair of attacker nodes may create a wormhole [32] and shortcut the normal flows between nodes. In the context of on-demand ad hoc routing protocols, the attackers may target the route maintenance process and advertise that an operational link is broken [33].

In addition to routing attacks, the adversary may launch attacks against packet forwarding operations as well. In MANETs, obtaining correct routing information does not guarantee that packets would reach their destinations correctly. There are two types of packet forwarding attacks: denial-of-service and selfishness. In a denial-of-service attack, the adversary sends an excessive number of junk packets into the network via the network layer. These packets waste a significant portion of the network resources, and introduce severe wireless channel contention and network congestion in the MANET. In selfishness, the attacker drops packets that it is supposed to forward in order to save its own resources. The issue of selfishness will be discussed in more details in the next section.

2.4.4 Selfish Nodes in MANETs

Selfish nodes, which is the main topic of this thesis, are defined as nodes that do not provide service to other nodes but at the same time request service from others. Such nodes are also called misbehaving nodes and their behaviour is termed selfishness or misbehaviour [34]. Unlike typical malicious nodes, the main objective of selfish nodes is to reserve their resources for their own use. Selfish nodes can be put into both passive and active attack groups. They are not actively attacking the network but have a negative effect on the communication efficiency. For example, in MANETs, the selfish nodes use the services offered by other nodes to forward their packets in the multi-hop communication but intentionally do not forward packets for others.

As stated earlier, in the absence of a centralised management station, the nodes in the multi-hop ad hoc network themselves have to act as routers. To send a packet to other nodes that are beyond its radio range, a node has to rely on adjacent nodes to relay its packet to the next node until the packet reaches the destination. This is not a problem if all of the respective nodes belong to the same authority in which they all must cooperate to achieve the same goal. However, ad hoc networks are increasingly being used in civilian application where the nodes belong to different individuals. In such cases, there is no guarantee that all of them would cooperate.

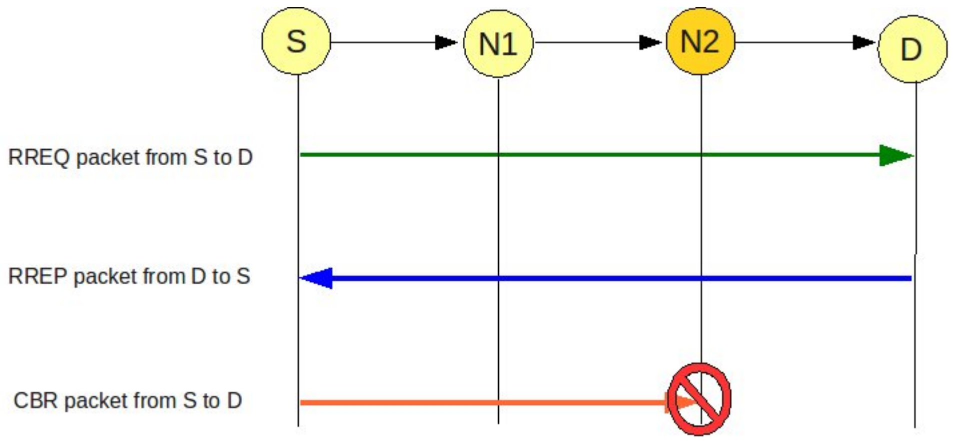


Figure 2.11: Misleading Node

In a MANET, nodes have to cope with several resources constraints such as energy consumption and storage capacity [35]. Cooperating by providing network services for other nodes such as relaying packets consumes energy. Nodes providing these services do not gain any direct benefit for their act. Therefore, there is no interest for nodes to participate in the routing and forwarding process. In fact, by refusing to cooperate in relaying packets, the nodes could reserve their limited resources for their own use and prolong their own life time. However, these misbehaving nodes should be detected as they can lead to a significant drop in the overall network performance as we will see later.

Selfish nodes can be categorized into two groups: misleading and free-riding. Misleading nodes participate in route discovery process and then refuse to forward data packet when they get the request. Figure 2.11 illustrates the behaviour of a misleading node. In this diagram the source node wishes to send a message to the destination node which is not in the neighbourhood. Since the source node does not have any route to the destination, it has to broadcast a ROUTE REQUEST control packet to the network. Each of the intermediate nodes including the misleading node (Node N2) cooperates by rebroadcasting the ROUTE REQUEST packet until it reaches the destination node. The destination node replies back with ROUTE

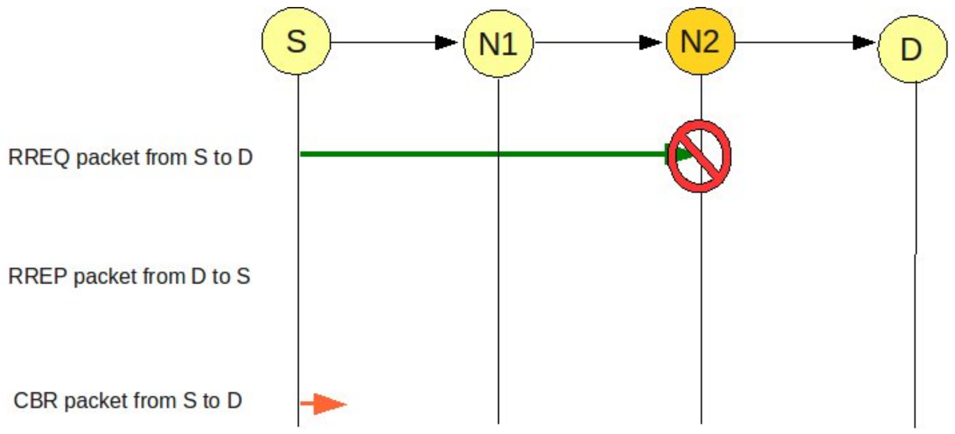


Figure 2.12: Free-Riding Node

REPLY control packet to the source node following the route by which it received the original ROUTE REQUEST packet. In this stage the misleading node still cooperates by forwarding the control packet to the next node. Once the source node receives the ROUTE REPLY packet, the route discovery process has completed and the source node now can start sending the data packet. However when the misleading node receives the data packet to forward to the next node, the data packet is silently dropped. One reason for choosing data packets to be dropped is because data packets are longer and require more energy to forward compared to control packets.

Free-riding nodes tend to save their energy to the maximum. They ignore all control and data packets that are not destined for them. As depicted in Figure 2.12, when the free-riding node (Node N2) receives a ROUTE REQUEST packet destined to the destination node, it ignores and drops the packet. If there is no other alternative route to the destination node, the source node will not receive a ROUTE REPLY packet and thus could not establish communication with the destination node. One important consequence of this action is free-riding nodes will excuse themselves from the route and will not receive data forwarding request. Misleading nodes are similar to free-riding nodes in such a way that both of them deny services when requested but expect other nodes to forward packets on their behalf in spite of their own selfishness.

2.5 Summary

In this chapter the basic review of wireless networks was introduced. Radio propagation models which explain the characteristics of radio signal were briefly discussed. Then, a description of MANETs and different communication technologies used in MANETs were presented. The major issues involving in routing protocols and different classifications of routing protocols for MANETs were described. Dynamic source routing (DSR) was explained in more detail as it is being used as routing protocol in our simulation model. Then, we provide a basic concept of network security. Security issues related to MANETs were discussed including classification of attacks. Finally, an introduction about selfish nodes, particularly selfish and misleading nodes, was presented. In the next chapter, we will discuss the related published work in detecting and dealing with those selfish nodes in MANETs.

Chapter 3

Trust and Related Work

3.1 Introduction

A Mobile ad hoc network (MANET) is a self-organised network. In such networks, there is no infrastructure, no central authority and no centralised trusted server. Critical network functions such as routing and packet forwarding are assumed to be carried out by each node which may act as both terminal and router. Such assumption is true if all of the nodes belongs to a single authority or company and pursue the same goals. However, in an open MANET, each node is its own authority domain. There is no guarantee that all nodes in the network will cooperate in carrying out those network functions. In order to mitigate security threats from misbehaving, nodes have to make their own judgement to make decision on which of their neighbours they think would be cooperative and which would not. Trust and reputation have been suggested as one of the solutions to security problem in routing and data forwarding. In a distributed environment, trust and reputation can be used to improve security, to support decision-making and to promote node collaboration. Before going into the mechanisms of detecting misbehaving and cooperation motivations, which is the main topic of this chapter, we will first focus on the notion of trust.

3.2 About Trust

In a general sense, the purpose of security mechanisms is to provide protection against malicious parties. In hard security mechanism, terms that were first described by Rasmussen and Jannson [36], we protect the resources from a malicious user by restricting access to the resources by using authentication and access control. Examples of this traditional security mechanism are cryptographic algorithms and firewalls. However, in the case of selfishness, the problem is reversed. We want to protect the users from those who offer resources. Social control mechanisms or soft security is more suitable to address this issue. Trust and reputation represent a soft security approach.

Trust is a concept we encounter in everyday life. Based on our experience in the physical world, we extract the necessary information that can help us to build trust, increase users' confidence and reduce the risk. It is a fundamental aspect on which we base our interactions, transactions and communications in our daily life. For example, in a job interview, an employer may conducting a short-time interview where the successful applicant may come to work on the very next day. Both of them have never met before and are stranger to each other. In this case, they both take a risk and must have basic trust between each other. The employer trusts that the new appointed employee would perform the job as in the description. On the other hand, the employee trusts that the salary will be paid at the end of the month.

There are many definitions and models for trust. According to Eschenauer et al. [37], trust is defined as *a set of relations among entities that participate in a protocol.* These relations are based on the evidence generated by the previous interactions of entities within a protocol. In general, if the interactions have been faithful to the protocol, then trust will accumulate between these entities. Based on the previous work described in [38, 39, 40], the notion of trust is briefly defined as: *trust is the* *degree of belief about the future behaviour of other entities, its calculation is based on the past experience with and the observation of the others related actions.*

The concept of trust can also be applied in the computer environment. Mobile ad hoc networks may contain many peer nodes. In an open MANET, there is no a priori trust and each node is a stranger to another. These nodes also need trust before they exchange information. The main goals of a trust-based system in MANET are as follows:

- Provide information that allows nodes to distinguish between trustworthy and non-trustworthy nodes
- Encourage nodes to be trustworthy
- Discourage participation of nodes that are untrustworthy
- Cope with any kind of observable misbehaviour
- Minimize the damage caused by insider attacks

3.3 Type of Observations

A natural way of acquiring trust is through observation. There are two types of observation based on the source: first-hand observation and second-hand observation. In first-hand observation, a node uses its own observation and experience to compute trust. First-hand observation can be further classified into two categories: personal experience and direct observation. Personal experience of a node refers to the information it gathers through one-to-one interaction with its neighbouring nodes. On the other hand, direct observation refers to the information a node gathers by observing the interactions among its neighbouring nodes without been involved in the interaction.

In second-hand observation, a node uses information or recommendation provided by peers in its neighbourhood. This allows nodes to learn from each other. The information may be published either on each predefined interval time or when an event of interest occurs. Second-hand observation is used because most of the time first-hand observation is not enough to compute trust. A node may encounter another node which it has no experience with. The final trust computation depends on the combination of both own experiences and the recommendations from the neighbouring nodes.

Second-hand observation makes the system vulnerable to false report attacks. This vulnerability can be reduced by limiting type of information that can be used, be it positive information or negative information. However, both strategies have their own drawbacks. When only positive information is used, nodes cannot share their bad experiences. Malicious nodes can collude, distributing good reports about each other. Similarly, when only negative is used, nodes cannot share their good experiences. Malicious nodes can launch a bad-mouth attack, spreading bad reports on good nodes.

Second-hand observation also has no benefit if such information is not distributed by nodes. Nodes may refuse to distribute reports because of the same reason misbehaving nodes do not relay packets for others. Sending information packets does not give direct benefit to the sender and it consumes resources such as battery energy and processing power. Another reason for not providing information is to avoid being labelled as misbehaving. Depending on the mechanism used to detect misbehaving, if the information provided by a node is different from other nodes in

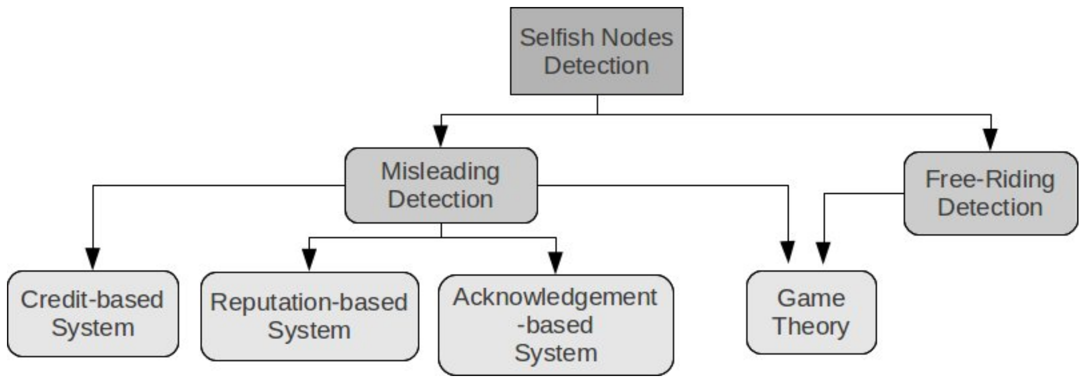


Figure 3.1: Categories of Selfish Node Detection Schemes

the network, the node may itself be accused of providing false information.

3.4 Related Works

As explained in the previous chapter, there are two types of selfish nodes: misleading and free-riding nodes. In this section, we describe some schemes that have been proposed to detect and mitigate the effect of misleading nodes in a mobile ad hoc network. Misleading node detection schemes can be broadly classified into three categories: credit-based systems, reputation-based systems and acknowledgement-based systems. Later, we also present some known detection schemes to detect free-riding nodes. A diagram with the categorization is shown in diagram 3.1. In this section, we explain briefly the main features of some schemes under each category.

3.4.1 Credit-Based Systems

Credit-based systems are designed to provide incentives for nodes to faithfully perform networking functions. To achieve this goal, virtual (electronic) currency or similar payment system is used. Nodes get paid for forwarding message for other nodes. When they request other nodes to forward their message, they use the same

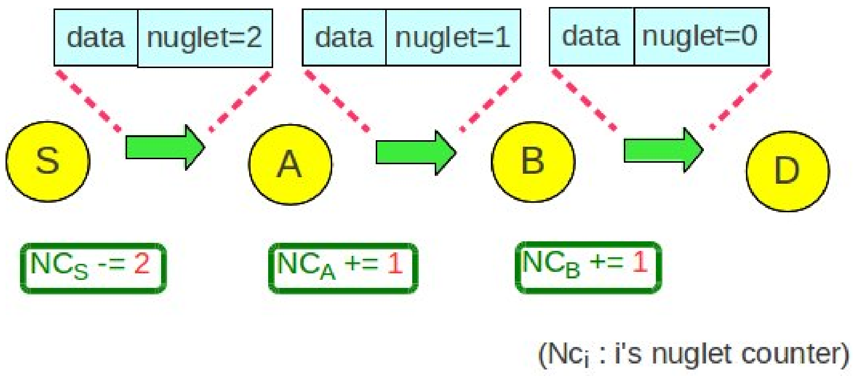


Figure 3.2: Packet Purse Model

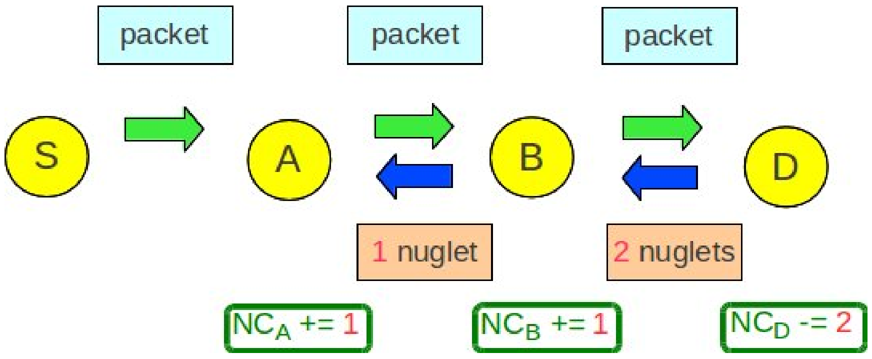


Figure 3.3: Packet Trade Model

payment system to pay for such services.

Buttyan and Hubaux [41] proposed a scheme in which a virtual currency called nuglets are used as payment for packet forwarding. In this scheme, the authors proposed two payment models: Packet Purse Model and Packet Trade Model. In Packet Purse Model, as depicted in Figure 3.2, the source of the packet pays by loading some nuglets into the packet before sending it. Each intermediate node takes some nuglets in return for forwarding the packet. However, in this model, the sender has to know the number of hops required for the packet to reach the destination so that sufficient nuglets would be loaded into the packet. If the packet runs out of nuglets, it will be discarded. The advantage of this model is that since the source has to pay for the packet, it discourages users from flooding the network.

In Packet Trade Model, as illustrated in Figure 3.3, the destination of a packet pays for the packet. In this model, the source does not need to know how many nuglets need to be loaded into the packet. To implement Packet Trade Model, each intermediate node buys a packet from its previous node for some nuglets and sells it to the next node for more nuglets. This way, every intermediate node earns some nuglets and the total cost of forwarding the packet is paid by the destination node. Disadvantage of this model is since the source does not have to pay for sending packets, a malicious node will be able to overload the network with useless data without being charged. In addition, the intermediate nodes could also take out nuglets from a packet and then deny the forwarding service. To implement either models, a tamper-proof hardware [41] is required at each node so that the correct amount of nuglets is deducted or credited.

Besides the nuglet approach, Buttyan and Hubaux also proposed another scheme based on a credit counter [42]. In this approach, each node maintains a counter called nuglet counter. The counter is decreased every time the node sends packets of its own and increased for each packets it forwards for other nodes. The counter has to be more than zero before a node is allowed to send its packet. Therefore it needs to help other nodes continuously. Similar to their previous approaches, this scheme also requires tamper-proof hardware modules to protect the counter from been increased illegally.

Zhong et al. proposed a cheat-proof, credit-based mechanism called Sprite [43]. Sprite releases the requirement of tamper-proof hardware at any node. In Sprite, nodes keep receipts of all the messages they receive and forward. Each receipt contains a hash of the received message. All these receipts would be reported to a centralised Credit Clearance Service (CSS) which then decides the charge and credit for the reporting nodes. This scheme can effectively stimulate cooperation among the nodes. However, the scheme relies heavily on the availability of the trusted CSS.

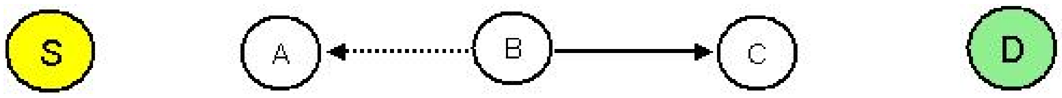


Figure 3.4: Watchdog

In a self-organised network, the requirement of central billing services in a system is not only inappropriate but also could become a single point of failure.

3.4.2 Reputation Based Systems

Reputation-based systems define a method for keeping track of a node's history by monitoring the node's past actions and then use this information to decide whether or not to include or exclude the node from the network. Reputation systems are commonly used in many area of electronic transactions such as eBay and Amazon. In wireless mobile ad hoc network, reputation mechanisms are applied to address threats arising from misbehaving nodes.

One of the earliest proposals to mitigate the routing misbehaviour are watchdog and pathrater. These two components were proposed by Marti et al. [44] to improve throughput in mobile ad hoc networks in the presence of misbehaving nodes. These components are implemented on top of DSR. Watchdog is the monitoring component which detects non-forwarding by overhearing the transmission of the next node. It is also the most common monitoring mechanism used in many reputation schemes.

Figure 3.4 illustrates how the watchdog works. In this example there exists a path from source node (S) to destination node (D) through intermediate nodes A, B and C. When a packet is sent to node A from S, node A cannot relay it all the way to node C. So it transmit the packet to node B with the intention that node B will forward the packet to node C. Since node B is in its radio range, node A can detect if node B performs the forwarding by overhearing the transmission. Node A will store a copy of the packet into its buffer and delete the packet when it overhears node B has forwarded the packet. If a packet stays in the buffer for longer than a certain period, the watchdog sees it as a misbehaving act and decreases the rating of the node responsible for the forwarding the packet. If the packet is forwarded successfully without any modification, the rating for the particular nodes will be raised. Each node takes part in the detection of the misbehaviour of its neighbour. Each node also has its own rating and it is updated periodically.

The second component in the proposal, Pathrater, maintains the rating for every other node that it knows in the network. It is used to mitigate the effect of misbehaving nodes when making route selection. The selection is based on the highest average node rating instead of just based on the shortest path. In this way, the misbehaving nodes would be less likely to be selected and thus improve the network throughput.

These two mechanisms are designed to mitigate the effect of a misbehaving nodes presence in a network. In a simulation, it is shown that the total throughput of the network is at an adequate level even when 40% of the participating nodes are misbehaving [44]. The disadvantage of this proposal is although the misbehaving nodes can be detected, they would not be punished but rather relieved from forwarding packets for others. This means that there is no motivation for the nodes to cooperate

The COllaborative REputation (CORE) mechanism was proposed by Michiardi and Molva [45] to enforce node cooperation in MANETs based on a collaborative monitoring technique. It uses a watchdog mechanism for monitoring but is complemented by a reputation mechanism to distinguish between subjective reputation (own observation), indirect reputation (positive reports by other nodes) and functional reputation (task-specific behaviour). Each node computes a global reputation value for every neighbour obtained as a result of the combination of these three different types of evaluations.

CORE consists of two basic components: the watchdog mechanism and a reputation table. The watchdog mechanism is used to detect misbehaving nodes. When a node forwards a packet, the node's watchdog verifies that that the next node in the path also forwards the packet. This is done by listening promiscuously to the next node's transmission. If the next node does not forward the packet after a certain time out, a negative value is assigned to the rating factor of that observation and the reputation rating of the provider will be decreased.

A subjective reputation is locally calculated directly from a node's observation. The node calculates the reputation of its neighbours that are in its transmission range. Indirect reputation is the second-hand information provided by other nodes in the network. It influences the final impression of a node. However, CORE puts a restriction on the type of information propagated by the nodes. To eliminate a denial-of-service attack through badmouthing where a malicious node spread negative reputation information on a benign node, only positive reputation values can be exchanged. Functional reputation is the combined value of subjective and indirect reputation of a node with respect to different functions. Different functions may be assigned different weight based on their importance. For example, data packet forwarding function may be seen to be more important than forwarding packets with route information. This means that data packet forwarding will be given greater weight in the reputation calculations.

In this scheme, nodes have to contribute on a continuous basis to remain trusted. Nodes that are not active will gradually have their reputation decreased to null. The purpose of this is to eliminate any advantage for nodes which stay idle for most of the time and only be active when they want to communicate. A node that has a reputation value less than the threshold would be denied service by other nodes.

As stated before, nodes are only allowed to exchange positive reputation information and thus can resist simple denial of service attacks. However, it does not address the issue of false praise attacks where two or more malicious nodes may collude and praise each other to increase their reputation ratings. One distinct characteristic of CORE is it gives consideration to the function when calculating the reputation. Nevertheless an adversary may be able to cover their misbehaviour with respect to one function and well behaved for other functions. The nodes may also be selective in choosing the functions for which they want to cooperate. For example, the node cooperates only on specific functions that do not consume lots of power and misbehaves on high power functions to save scarce resources.

OCEAN, Observation-based Cooperation Enforcement in Ad hoc Networks is an extension to the Dynamic Source Routing (DSR). This protocol is proposed by Bansal and Baker who focus on the robustness of packet forwarding at the routing layer [46]. OCEAN aims at having simple reputation mechanism by avoiding complex second-hand reputation information management such as false accusation and trust relationships. In doing so, OCEAN relies exclusively on first-hand observation. Positive observation increases the reputation rating and vice versa. Therefore, OCEAN can be considered as a stand-alone architecture.

To detect and mitigate the effect of misleading nodes, OCEAN uses of five components: NeighborWatch, RouteRanker, Rank-Based Routing, Malicious Traffic Rejection and Second Chance Mechanism. These components may reside on each node in the network. NeighborWatch monitors the behaviour of the neighbour of a node through promiscuous observations. Whenever it requires its neighbour to forward a packet, it makes a copy of the packet checksum into a buffer. If its neighbour does not forward the packet within a given time, NeighborWatch registers a negative event and removes the checksum from the buffer. On the other hand, if its neighbour attempts to forward the packet, the module compares the packet with the one in the buffer. If it matches, the module registers a positive event and removes the checksum from the buffer. If it does not match, the module assumes that the packet has not been forwarded.

RouteRanker maintains the ratings for each of the neighbouring nodes. A new node that has just joined the network is assigned a neutral rating which is zero. OCEAN takes both positive and negative events registered from NeighborWatch module to calculate the total rating. However the absolute value of the negative decrement is larger than the positive increment. Once the rating of a node falls below a certain threshold, the node will be added to a faulty list. This list represents the list of observed misbehaving nodes. A route is considered good or bad depending on whether the next hop is on the faulty list.

Rank-Based Routing is responsible for selecting routes by using information from NeighborWatch. To avoid selecting a route containing nodes in the faulty list, an additional field, called the avoid list, is added to the DSR Route-Request Packet (RREQ). The avoid list represents the list of nodes that the RREQ transmitter wishes to avoid in its future routes. After receiving an RREQ packet, a node check the RREQ avoid list to avoid the misbehaving nodes when re-broadcast the RREQ. The node also appends its own faulty list to the avoid list of the RREQ packet.

The Malicious Traffic Rejection module is used to reject traffic from nodes that are considered misbehaving. In OCEAN, all traffic from a misbehaving node is rejected. As a result, a misbehaving node could not relay its own traffic under the guise of forwarding it on for others.

Second Chance Mechanism gives an opportunity to nodes that were previously considered misbehaving to become operational again. A timeout approach is used where a misbehaving node is removed from the faulty list after a fixed period of

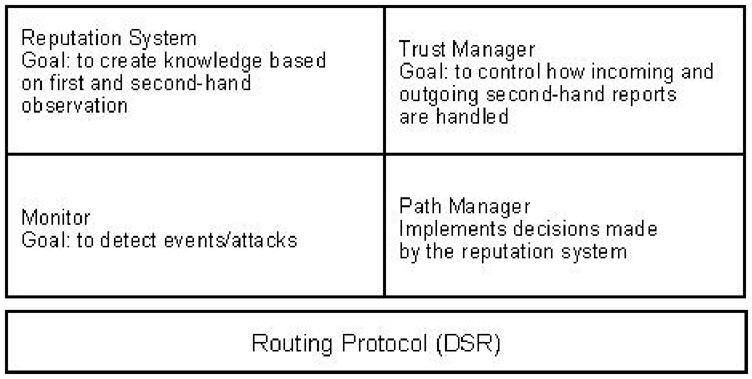

Reputation System
Goal: to create knowledge based on first and second-hand observation
Trust Manager
Goal: to control how incoming and outgoing second-hand reports are handled
Monitor
Goal: to detect events/attacks
Path Manager
Implements decisions made by the reputation system
Routing Protocol (DSR)

Figure 3.5: CONFIDANT

inactivity. However, its rating is not increased to neutral. If the node continues the misbehaviour, it can easily be added back to the faulty list.

OCEAN, unlike many others reputation schemes, does not consider second hand information. Nevertheless, the authors claim that OCEAN performs almost as well and sometimes in some environment better than a generic scheme that uses second-hand information. For example, since OCEAN only uses its own direct observation with the neighbours, it is more resilient to false report attacks. Simulations also have shown that with a low faulty threshold, OCEAN can perform better. In a highly mobile network, OCEAN can perform reasonably well without the need to exchange second hand information.

Another proposed reputation mechanism is called CONFIDANT (Cooperation Of Nodes: Fairness in Dynamic Ad-hoc NeTworks)[47]. The mechanism which is introduced by Buchegger and Boudec aims at detecting and isolating uncooperative nodes, thus making it unattractive to deny cooperation. It works as an extension to on demand routing protocols. With CONFIDANT, as depicted in Figure 3.5, each node has the following four main components, a monitor, a reputation system, a path manager and a trust manager. These components interact with each other to provide and process protocol information. A monitor is used to observe the behaviour of the neighbours by listening to the transmission of the next node. If any suspicious events are detected, an ALARM message will be send to the trust manager. A trust manager is used to manage ALARM messages. It makes decisions about providing or accepting route information, accepting a node as part of a route or taking part in a route originated by another node. The reputation system is used to rate every node in a network. A path manager is responsible for ranking a path according to the reputation of the nodes in the path. It also deletes paths containing misbehaving nodes from the path cache and penalizes the misbehaving nodes by denying all services to them.

3.4.3 Acknowledgement-Based Systems

Acknowledgement-based systems rely on the reception of acknowledgements to verify that a data packet was forwarded to the next hop. The basic idea is when a node forwards a data packet successfully over the next hop, the destination node of the next-hop link will send back a special two-hop acknowledgement to indicate that the data packet has been received successfully. Unlike reputation-based systems which use an overhearing technique, the nodes do not have to operate in promiscuous mode.

The first detection scheme under this category is proposed by Balakrishnan et al. called TWOACK [48]. The TWOACK scheme allows a node to know whether or not the node two hops away has received a data packet. Figure 3.6 illustrates the operational details of the TWOACK scheme. Suppose that the process of Route Discovery has already yielded a source route (S->NI->N2->D) from a source node S to destination node D. Every time a node sends or forwards a packet, it stores the packet ID in a table. When N2 receives a data packet, it sends out a special two-hop acknowledgement packet called TWOACK over two hops back to node S,

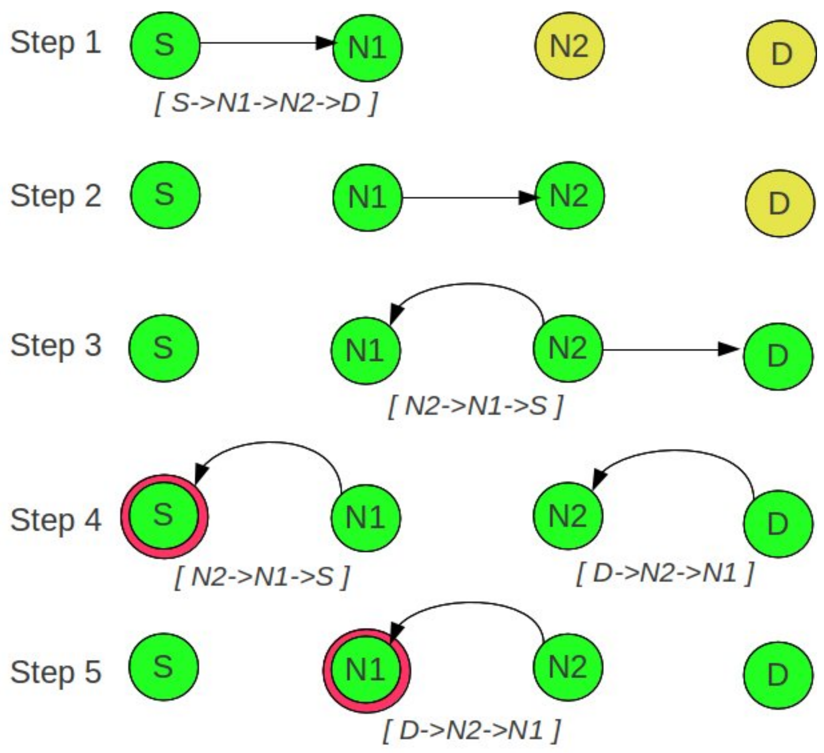


Figure 3.6: TWOACK Scheme

carrying the packet ID of the corresponding received data packet. Upon receiving the TWOACK packet, node S removes the packet ID from its table. Every set of three consecutive nodes along the source route will do the same procedure. If node S does not receive such acknowledgement within a predefined time, the current link (N1->N2) is considered misbehaving. The routing protocol will avoid the accused link in all future routes, resulting in an improved overall throughput performance for the network. This scheme can be implemented on top of any source routing protocol such as DSR.

One main drawback of TWOACK is that the scheme uses an excessive number of TWOACK packets. To help reduce the overhead, the authors also propose another scheme called S-TWOACK (Selective-TWOACK). In S-TWOACK scheme, instead of sending back a TWOACK packet every time a data packet is received, a node waits until a certain number of data packets has arrived. The node then sends back one TWOACK packet acknowledging multiple data packets that have been received so far.

In [49] the authors introduce a revised version of TWOACK called 2ACK. 2ACK was proposed to served as an add-on technique for the DSR routing protocol to detect misbehaving links and mitigate their effects. It is different from the previous schemes (TWOACK and S-TWOACK) in terms of the following:

1. The receiving node in the 2ACK scheme only sends acknowledgement packets for a fraction of received data packets, while in the TWOACK scheme, the acknowledgement packets are sent for every data packet received. This would reduce additional routing overhead in the scheme.
2. The 2ACK scheme provides an authentication mechanism to protect the acknowledgement packets from being fabricated.
3. Each acknowledgement packet in the S-TWOACK scheme acknowledges the receipt of a number of data packets but an acknowledgement packet in the 2ACK scheme only acknowledges one data packet.

The schemes mentioned above can only detect link misbehaviour. The schemes can not decide which node associated with the link is misbehaving. This is mainly due to the fact that communication takes place between two nodes and is not the sole effort of a single node. When a link is found misbehaving, it could be caused by either one of the following two scenarios:

1. The intermediate node does not forward the data packet or the acknowledgement packet to the next hop.

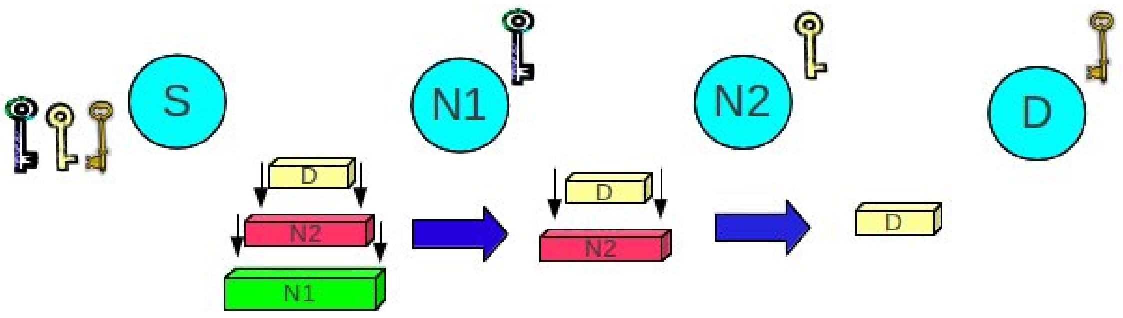


Figure 3.7: Onion Encryption

2. Upon receiving the data packet, the node does not send back the acknowledgement packet.

Awerbuch et al. [50] describe a mechanism to detect misbehaving nodes by checking the behaviour of links around that node. The mechanism uses a probing technique and acknowledgements not only by the destination but also by intermediate nodes. It uses onion encryption to embed a probe command for a specific node into normal data packets.

In onion encryption, as illustrated in Figure 3.7, a source node that wishes to send a message, wraps the message with several layers of encryption, one for each node using the shared key, and sends it through the intermediate nodes. Onion-style encryption ensures that the content of the each layer of message can only be read by the intended recipient. For example, N1 can only read the message that was encrypted with its key before sending the remaining packet to the next node and so on.

Finally, when D decrypts its onion layer, it will find the probe command and send back an acknowledgement packet to the source. As soon as an acknowledgement is missing, the source node starts a binary search in the path to find out where packets are being dropped. The source node simply sends probes to the selected nodes and

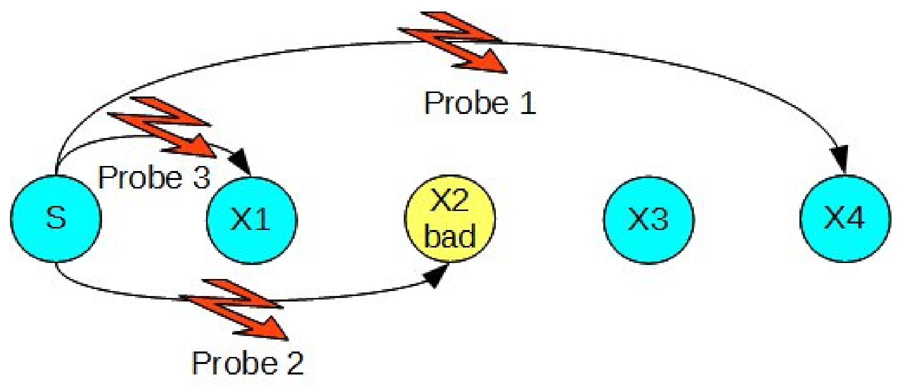


Figure 3.8: Binary Probing

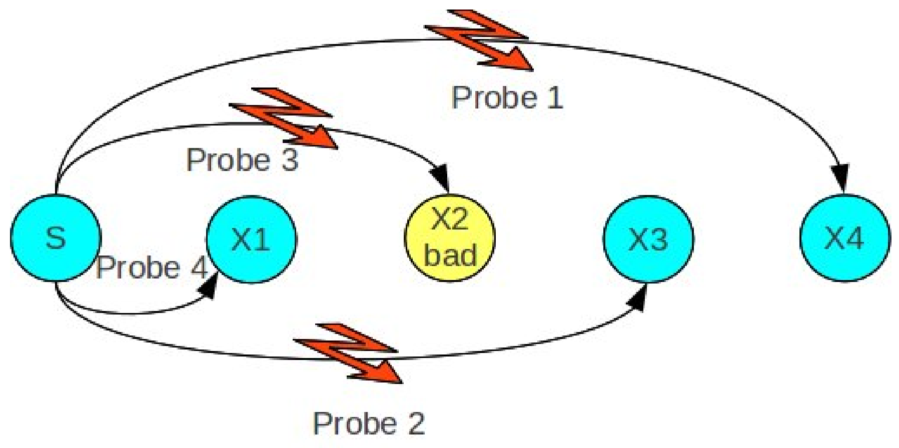


Figure 3.9: Iterative Probing

waits for their replies. Figure 3.8 shows how the binary probing works.

This approach has several drawbacks. The onion-encryption is relatively expensive. The sender has to encrypt probe packet multiple times depending on the length of the path. There is also no reliable detection of misbehaving nodes. If the misbehaving node gets the probe packet, it will realise straight away that a probing is under way. It can decide to temporarily cooperate until the probe is over and then continue to drop packets. In addition, it may be able to selectively drop probe packets meant for another node. The nodes then would not acknowledge the probe and would be wrongly accused as misbehaving.

Kargl et al. [51] introduce another approach called iterative probing. As illustrated in Figure 3.9, the sender will send a probe to the destination when it has not received an acknowledgement packet for a certain amount of time. If there is no reply within a certain timeout, it will send a probe to Xn-1 and so on until it receives a reply from a node or reaches X1. In this approach, selfish node only could notice that there is ongoing probing when it is its turn to answer a probe. So the selfish node will be able to drop probe packets sent to nodes earlier in the path.

However, iterative probing still could not absolutely decide which node in the link is misbehaving. The misbehaving node (X2 in the diagram) could check if it is a probe addressed to itself and decide to reply to the probe packet. In this case, the next node (X3) could be accused of misbehaving. In order to overcome this dilemma, the authors also present another technique called unambiguous-probing. In this approach, they combine the iterative probing with overhearing. Node X1 will be asked to check if it can overhear the forwarding of a following probe packet by node X2. If this probe fails and X1 do not overhear X2 forwarding the packet, then it can be determined that X2 is dropping the packet. Otherwise, X3 is the misbehaving node in the link.

All of the previous proposals discussed so far are used to detect misleading nodes. As a reminder, misleading nodes participate in route discovery process but then drop the data packet when they receive the request. Free-riding nodes refuse to get involved in the route discovery process and as a result would not get any request to forward data packets. Since free-riding nodes are not expected to forward data packets, the techniques which are described so far are not effective.

3.4.4 Game Theory

Game theory is a branch of economics that uses models to study interactions with formalized incentive structures ("games"). It has been applied to many fields of study, including policy on taxation, design of roads and development of transportation networks. Recently, the use of game theory has been extended to a variety of areas in communication networks including packet forwarding.

Games theory provides tools to study situations of conflict and cooperation. Such a situation exists when two or more decision makers (also referred as players) who have different objectives act on the same system or share the same set of resources. Therefore, game theory is concerned with finding the best actions for individual decision makers in such situations and recognizing stable outcomes. The assumptions made while formulating a game are [52]:

1. There are at least two players in a game and each player has available to him/her two or more well-specified choices or sequences of choices.

2. Every possible combination of plays available to the players leads to a well-defined end-state (win, lose or draw) that terminates the game.

3. Associated with each possible outcome of the game is a collection of numerical payoffs, one to each player. These payoffs represent the value of the outcome to the different players.

4. All decision makers are rational; that is, each player, given two alternatives, will select the one that yields the greater payoff.

In a multi-hop MANET, the establishment of routes relies on the intermediate nodes forwarding packets for each other. In a scenario where the node are not controlled by a single authority, an intermediate node has the option to either participate or not participate in a forwarding process. The network operation of forwarding packet can therefore be modeled as a game and the nodes in the network are the players of this forwarding game. Since forwarding packets for others consumes energy, the best strategy for any node in such a MANET is not to cooperate in packet forwarding. However, if all nodes decide to behave in this way, as we will see later on, the network will collapse.

In game theoretic terms, there is a set of strategies. Since nodes are self-interested and rational, their main focus is to increase their own profits. They will follow a particular strategy unless they are convinced that they can benefit more by following some other strategy. The objective of game theory is to look for the Nash equilibrium point [53] where a player cannot increase his or her payoff by changing strategies while other players' strategies remain fixed. Such a game theory approach can be used to deal with both misleading and free-riding nodes.

Generous Tit-For-Tat (GTFT) and multiple-GTFT (m-GTFT) [54] are the first algorithms to use game theory in MANETs. GTFT applies to the case where all requests are relayed by just one node to reach their destinations. m-GTFT is for when multiple relays exist between the source and the destination. These algorithms allow a node to balance the energy consumed for other nodes with the energy used by others for itself; and to find an optimal trade-off between blocking probability and power consumption.

Michiardi et al. [45] proposed two methods to analyze the CORE algorithm from the perspective of a cooperative game approach and a non-cooperative game approach. In a cooperative game, players reach an agreement through communication,

Table 3.1: Prisoners Dilemma

P1\P2	Confess	Not confess
Confess	(5, 5)	(0, 10)
Not confess	(10, 0)	(1, 1)

while in a non-cooperative game, all players pursue their own profits independently. The most famous example of a non-cooperative game is the Prisoners Dilemma [55]. Table 3.1 shows the terms for prisoners P1 and P2 if each of them confesses or not. While being separated in different rooms and not aware of the each other's strategy, confessing is the best strategy for the individual prisoners. Thus, "confess-confess" is the Nash equilibrium. On the other hand, if they can cooperate, both prisoners can get the best profit.

Mahajan et al. [56] proposed a scheme called CATCH which attacks both the connectivity and forwarding avoidance problems of misbehaving nodes at the same time. The scheme is essentially build on top of the watchdog scheme in [44]. It uses an existing majority of cooperative nodes to discourage free-riding. To achieve this, CATCH uses anonymous messages (in which the sender's identity is hidden) to tackle two critical problems. First, CATCH allows a cooperative node to determine whether its neighbouring nodes are free-riding. Second, it enables the cooperative neighbours of a free-riding node to disconnect it from the rest of the network. Using game theory, the authors attempt to prove that CATCH induces cooperation to be an evolutionary stable strategy.

3.4.5 Free-Riding Detection Scheme

There are a few mechanisms which specifically designed to detect free-riding nodes. Huang et al. [57] proposed a distributed detection and punishment algorithm for free-riding dropping in route discovery. The basic idea is that when any node receives a RREQ packet, it should either rebroadcast it or send a RREP back if its TTL is not zero. In the paper, the authors neglect the number of RREP replies in the algorithm. They claim that for networks of random topology and medium scale, the number of RREP is much smaller than that of RREQ. The monitoring nodes will calculate the number of RREQ relayed by each of their neighbouring nodes and compare it with the sum of itself. Those which relay approximately the same number of RREQs as the monitoring node itself are considered cooperative. Those which relay many fewer RREQs are regarded as free-riding.

Paul and Westhoff introduced the Context Aware Scheme [58] which uses unkeyed hash chains and a promiscuous mode to detect the misbehaviour during the route discovery phase. The observers of misbehaviour independently communicate their accusation to the source. To convict a culprit, more than three accusations are needed. If there is only one accusing node, the accusing node itself will be considered to be an attacker. The drawback of this scheme is that it is more beneficial for a node not to send the alarm message to avoid the risk being the only accuser and regarded as attacker.

3.5 Summary

In this chapter, trust is presented as a solution in mitigating security problem in routing and data forwarding. The definition and measurements of trust are also discussed. We indicate that trust can be acquired by doing observations. Two types of observations based on the source are discussed along with their pros and cons. A comprehensive review of research work focusing on detecting misbehaving nodes in MANETs has been presented. The related work is categorized into three groups: credit-based systems, reputation-based systems and acknowledgement-based systems. Finally we discussed a few published works designed to detect free-riding nodes instead of misleading nodes targeted by other previous work.

Chapter 4

Effects of Misbehaving Nodes

4.1 Introduction

The performance of a MANET will be affected by many factors. In this chapter, we show the effects of playground area size and the mobility speed of the nodes that could influence the performance. This will give us more understanding about the characteristics of a MANET.

Lack of central administration means that all participating nodes have to cooperate with each other to perform the necessary network functionalities that are usually provided by a centralised station in a traditional infrastructure-based network. In an open MANET, the nodes may belong to different authorities which could have different interests and objectives. Providing network services such as forwarding packets and detecting routes for others consumes network bandwidth, local CPU time, memory and battery power which are limited in MANET nodes. Therefore, some of the nodes may refuse to cooperate.

The issues of non-cooperative or misbehaving nodes in a MANET is a popular topic in the research area. In this chapter, we want to discuss the effects of these misbehaving nodes (misleading and free-riding) to the performance of a mobile ad hoc network. Firstly, we need to have an understanding of the simulation tools that we are using. The following section serves as a short introduction on OMNET++.

4.2 The Discrete Event Simulation System - OMNET++

All simulations results in this thesis were run on OMNET++ version 4. The name OMNET++ stands for Objective Modular Network Testbed in C++ [59]. It is a public-source, component based, modular, open architecture and discrete event simulation environment with strong GUI support. OMNET++ has been publicly available since 1997 and can be used under the Academic Public License which makes the software free of charge for non-profit use. OMNET++ is available on all common platforms including Linux, Mac OS/X and Win32 platform (XP and 2000). It offers a command line interface as well as a powerful graphical user interface. The primary application area of OMNET++ is the simulation of communication networks. It can also be used to model other areas such as simulation of complex IT systems, queuing networks and hardware architecture.

An OMNET++ model consists of hierarchically nested modules which communicate by passing message to each other. OMNET++ models are often referred to as networks. The top level module is the system module. The system module contains sub-modules which can also contain sub-modules themselves. The depth of module nesting is not limited. This allows the user to reflect the logical structure of the actual system.

OMNET++ provides the basic machinery and tools to write simulation components for computer networks, queuing networks and other domains. However, specific application areas are supported by various simulation models and frameworks. These models are developed completely independently of OMNET++ and follow their own release cycles.

There are already several frameworks available that can be used within OMNET++. These models are developed by various individuals and research groups for several specific areas. In our simulation, we use INET Framework [60], which is an open-source communication network simulation package that support wired, wireless and ad hoc networks. The INET Framework contains IPv4, IPv6, TCP, SCTP, UDP protocol implementations, and several application models. The framework also includes an MPLS model with RSVP-TE and LDP signalling. Link-layer models are PPP, Ethernet and 802.11. In addition, we make use of INETMANET [61] which extends the INET Framework with MANET protocols such as AODV, DSDV, DSR, DYMO and OLSR.

4.3 Effect of Playground Area in Performance of a MANET

In this section, we investigate the performance of a MANET when different playground areas are used. The simulated network contained 20 nodes distributed randomly in selected playground area sizes. In the simulation, each node is both a source and destination node. Each source node randomly select a destination node in the network for every data the source node sends. In this simulation, CBR traffic is used where the source node sends data packets with a frequency of 5 packets per second. CBR traffic effectively stresses a network because there are no control mechanisms to consider when flows are delayed or packets lost. Each node has a transmission

Table 4.1: Simulation parameters - Playground Areas

parameters	Value
Simulation Time	100 seconds
Speed	2, 5, 10 and 20 metres/second
Number of nodes	20
Packet Size	512 bytes
Network Area	200 - 1000 metres square
Mobility Model	Random Waypoint
Transmission Range	250 metres
Routing Protocol	Dynamic Source Routing (DSR)
Data Rate	5 packets/second
Traffic Type	CBR

range of 250 metres. The transmission range is based on the commercial Lucent WaveLAN radio interface. Each simulation experiment lasted for 100 seconds. All traffic is generated and the statistical data are collected after a warm-up time of 10 seconds in order to give the nodes sufficient time to finish the initialization process. The mobility process model implements a random waypoint model [24]. The random waypoint mobility model is a widely used model when evaluating MANETs [62, 3, 63, 64, 65, 66, 67] and hence is considered in this thesis. In the random waypoint mobility model, the nodes move to a random destination and stay at the destination for a while (Pause Time) before moving to another random destination within the playground area. In the simulation, we set the Pause Time to 0.1 second.

We also set the playground area sizes ranging from 200 metres square to 1000 metres square. We also change the nodes' speed to 2, 5, 10 and 20 metres per second to evaluate the effects of these parameters to the performance of the network. The summary of the simulation parameters used is shown in Table 4.1. The results presented for each value are the average of 10 simulation runs, each with different seeds. In this scenario, there is no misbehaving node in the network. All nodes cooperatively follow the routing protocol set which is dynamic source routing.

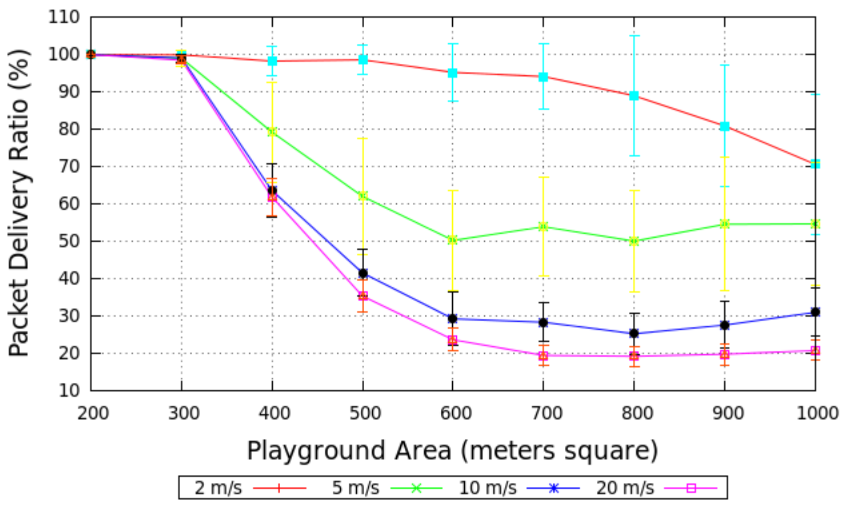


Figure 4.1: PDR vs Playground Areas

In evaluating the effect of playground area size and the node's mobility speed, we use the following performance metric.

4.3.1 Packet delivery ratio

This refers to the ratio of the total number of data packets that reach the receiver (destination node) to the total number of data packets sent by the source node. This performance metric is used to determine the efficiency and accuracy of the MANET's routing protocol. For a network, it is required that the packet delivery ratio is at a high level.

From the graph, we can see that packet delivery ratio decreases when we increase the size of the playground area. When the area size is big, the path length between any two connected nodes is longer than when the area is small. Routes are more prone to disconnections when the path length is longer in mobile networks. Any link failure along the route will result in the loss of a data packet. The mobility speed of the nodes is also another factor in link failure. When the nodes are moving with a higher speed, the possibility for any two connected nodes to be out of transmission range is greater than when the nodes are moving at a lower speed. As a result, when we increase the mobility speed of the nodes, the packet delivery ratio decreases. However, we also notice that packet delivery ratio is approximately the same when small playground area sizes are used (200 and 300 metres square), regardless of the nodes' speed. When the playground area is small, most of the nodes would be within transmission range of each other. Even when the nodes move with higher speed, there would be no requirement for multi-hop communications. Each of the nodes will be able to deliver data packets directly to the destination node without assistance from intermediate nodes.

4.4 Misbehaving Nodes Model

In our simulation, we make modifications to the DSR implementation in the INET-MANET Framework. In OMNET++, each node is identified by an unique id which starts with number zero. In our modification, we introduce misbehaving nodes in the network. The parameters used to indicate the misbehaviour are *BadHost* and *SelfishTypeNum*. *BadHosts* is an array containing the node id of the nodes that are misbehaving. *SelfishTypeNum* indicates the type of misbehaviour the nodes in the list exhibit. The value of parameter *SelfishTypeNum* is 1 for misleading nodes and to 2 for free-riding nodes. Both parameters are assigned in an initialization file for OMNET++ which is typically named *omnetpp.ini*.

Figure 4.2 shows the process flow when a misleading node receives an incoming packet. Initially, the node will check if the incoming packet is for itself. Even though a node is misbehaving, it should not compromise anything that gives benefit to itself

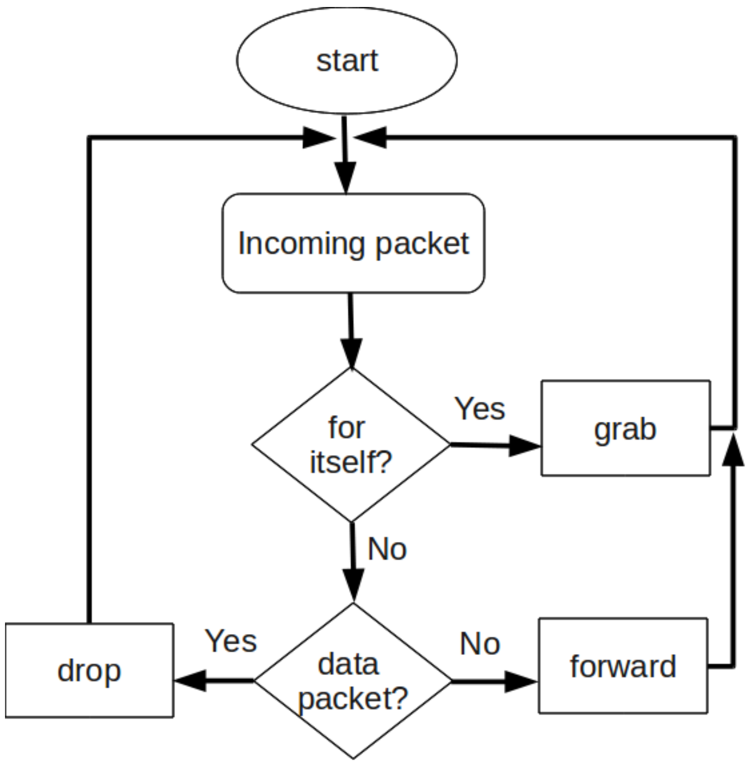


Figure 4.2: Misleading Nodes Process Flow

such as receiving packets meant for itself. This can be easily done for unicast packet such as RREP control packets and data packets. A node can compare its own address with the packet destination field in the packet header. If the packet is for itself, the node will simply grab it and send it to the upper layer. Otherwise, a misleading node forwards control packets and drops data packets destined for other nodes.

Figure 4.3 illustrates the process flow for a free-riding node. It looks similar to the process flow for misleading nodes except that selfish nodes also drop control packets meant for other nodes. To handle RREQ control packets, the program has to do extra work. The RREQ control packet is a broadcast packet and the destination field is always set to all ones. Whenever a node receives a packet that has destination address field different from itself, it must first check if the packet is a data packet or control packet. A data packet is different from control packet in term of the payload

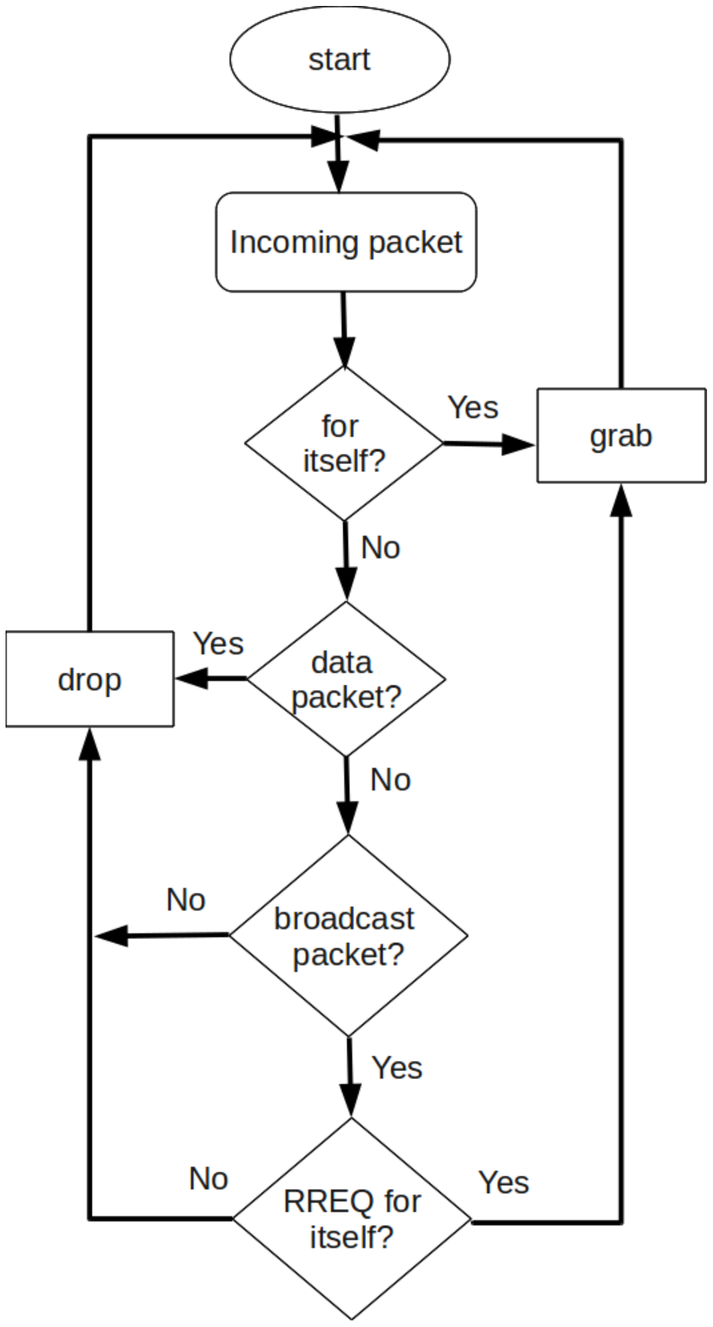


Figure 4.3: Free-Riding Nodes Process Flow

Table 4.2: Simulation parameters for Misbehaving Models

parameters	Value
Simulation Time	100 seconds
Number of nodes	20
Packet Size	512 bytes
Network Area	500m x 500m
Mobility Model	Random Waypoint
Transmission Range	250 metres
Routing Protocol	Dynamic Source Routing (DSR)
Data Rate	5 packets/second
Transport Protocol	UDP

size. The payload size for a control packet is always zero. For a control packet, the program has to check if the packet is a RREQ control packet. In DSR, the only broadcast control packet is RREQ. Once the program recognises that the incoming packet is a RREQ packet, the program will then check if the receiving node is the target of the RREQ. If the RREQ packet is destined to the node, it should grab the packet and pass the control packet to the upper layer for further process. Otherwise, the RREQ packet will be dropped.

4.5 Simulation Environment

Table 4.2 shows the parameters used in our OMNET++ simulations. A 20 mobile node network in a field size of 500m x 500m was used. The simulation time is 100 seconds. The mobile nodes move within the network space according to the random waypoint mobility model [24] with the speed of 2m/s. In the random waypoint mobility model, each node moves to a random location within the specified network area. Once the node reaches the target location, it will remain in that position for a time (pause time) before moving to another random location. In our simulation, the pause time is set to 0.1 second. The communication patterns used are constant bit rate (CBR) connections with a data rate of 5 packets per second. 20 connections are established at random so that each node has the chance to connect to every other node. The number of misbehaving nodes was increased for each simulation and takes values ranging from 0 to 20. To build confidence in our results, the experiments are repeated ten times with different seeds for each number of misbehaving nodes in the simulated mobile ad hoc network.

4.6 Simulation Metrics

For evaluating the effect of misbehaving nodes in a mobile ad hoc network, we consider the following metrics:

1. Packet Delivery Ratio: the ratio of total number of data packets received over the total number of data packets send in the application layer.
2. Mean Delay: the mean delay of the transmission time of each packet from the source to the destination.

4.7 Simulation Results

The results were collected as average values over ten runs of each simulation setting. The results are shown in Figures 4.4 and 4.5.

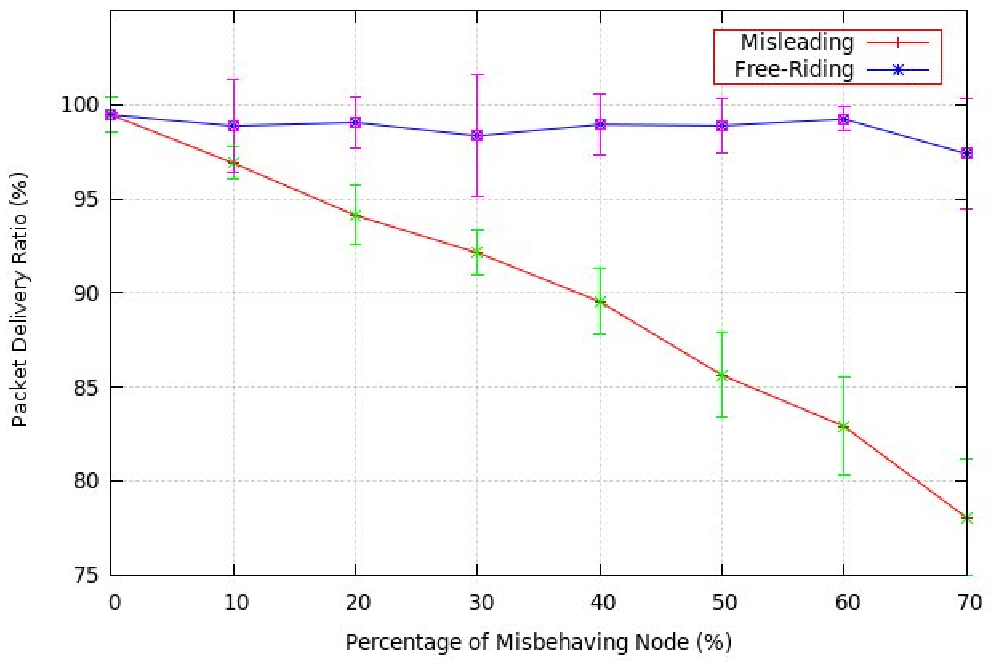


Figure 4.4: PDR vs Percentage of Misbehaving Nodes

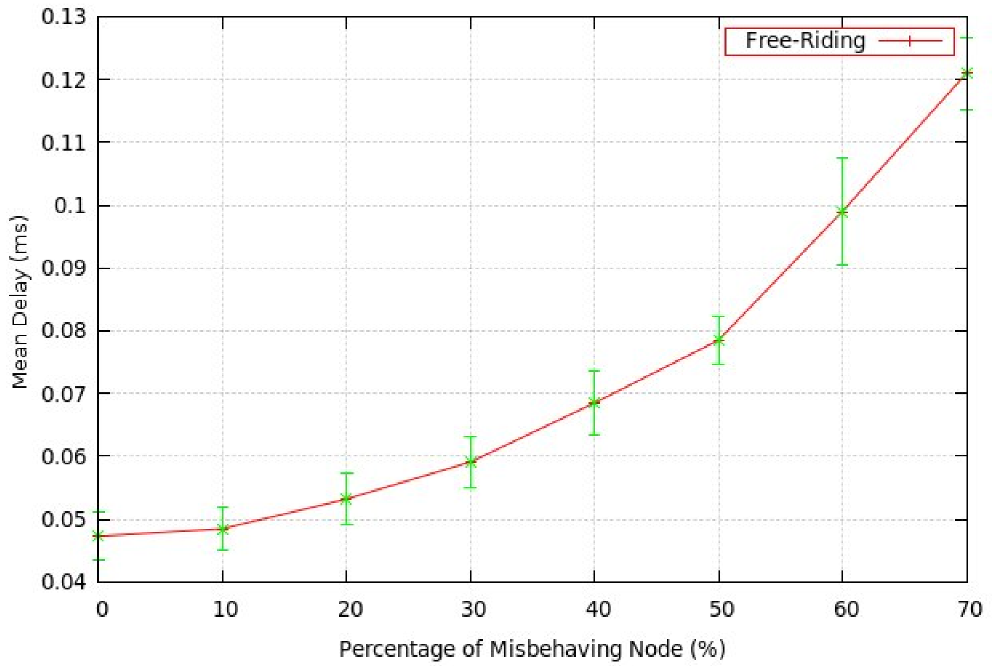


Figure 4.5: Mean Delay vs Percentage of Misbehaving Nodes

As we can see, as the number of misleading nodes increases in the network, the packet delivery ratio decreases. Figure 4.4 shows that the packet delivery ratio (PDR) degrades by 12% when half of the nodes in the network are misleading nodes. The decreasing rate of the ratio is not constant with the increased number of the misleading nodes. This is because some nodes are more crucial than others, which depends on their location. For example, nodes that are located at the edge of the network do not really affect the packet delivery ratio as they are not likely to be required to forward data packets.

Figure 4.4 also shows that the packet delivery ratio is not really affected by the increased number of free-riding nodes in the network. Free-riding nodes do not participate in the route discovery process and therefore they are not expected to forward any data packets. However, free-riding nodes cause an increase in mean end-to-end delay for the packets to reach their destination. Figure 4.5 illustrates that the mean end-to-end delay increases by 66% when half of the nodes become free-riding nodes. As the number of free-riding nodes is increased, the source node will have fewer options on which route the packets should travel. As a result, less attractive routes have to be chosen which means longer delays. It also means that the remaining cooperative nodes have to take an extra burden of forwarding packets. As the load increases, the delay also increases. The mean end-to-end delay for misleading nodes is not shown as they are not the main focus of this thesis.

4.8 Summary and Conclusion

Misleading and free-riding nodes degrade overall network performance and pose a serious threat to multihop routing in MANETs. In this chapter, we initially gave short introduction about OMNET++, the simulation program we use in the whole research project. Using OMNET++, we investigated the effect of playground area size and nodes' mobility speed to the performance of a MANET. The results show that when the playground area size or mobility speed is increased, packet delivery ratio (PDR) decreased. Finally, we concluded that the presence of misbehaving nodes (misleading and free-riding) in a MANET degrades the overall performance of the network. Misleading nodes only drop data packets to be forwarded and this affects the overall packet delivery ratio. Free-riding nodes only drop control packets to be forwarded and that prevents them from been requested to forward data packets.

Chapter 5

Detecting Free-Riding Nodes in MANETs

5.1 Introduction

In the previous chapter, the effects of misleading and free-riding nodes to the performance of a MANET have been discussed. Misleading nodes refuse to forward data packets for others and cause the number of data packets successfully delivered to the destination (Packet Delivery Ratio) to decrease. Free-riding nodes on the other hand have little effect to the packet delivery ratio. However if all nodes in the network are free-riding, no multi-hop communication could be establish and the performance of the network would be degraded. Unlike malicious nodes, these types of misbehaving nodes do not have harmful intentions towards the network although their actions may adversely affect the the performance of the network.

The main incentive for being misbehaving nodes is to save energy for their own communication. Among various resources associated with mobile nodes, data transmission is the most expensive function in the MANET environment. To send a bit over 10 or 100 metres distance, mobile nodes consume energy that can perform thousands to millions of arithmetic operations [68].

We argue that if any node merely intends to preserve own resources, it is easier for the node to become free-riding node, ignoring all packets (data and control) that are not destined for it. By becoming free-riding nodes, they can excuse themselves from being requested to forward packets for others but then still will be able to benefit from global connectivity offered by other nodes.

Most of the current mechanisms are designed to detect misleading nodes and are not effective to be used for detecting free-riding nodes. The main reason is those mechanisms monitor the data packet forwarded by the next hop and make a judgement from that. Free-riding nodes do not participate in the route discovery process and thus would not be used to forward data packets. Furthermore, some well-behaving nodes in the network in certain scenarios might not be required to forward data packets. Examples of those scenarios are:

1. The node is located at the edge of the network. At that location, the node does not have any other node to forward data packets to.
2. The network is already matured where all routing to every possible destination has been established. A new node then enters the network and wishes to use the network to send its data to another node. As long as there is no link break, there would be no changes in the routing table. The new node would not get any RREQ packet. As a result, the new node would not be required to do data forwarding.

In this thesis, we use the notation 'monitoring node' when referring to a node that is conducting the observation. We also use the notation 'monitored node' when referring to a node that is being observed. In other words, monitoring nodes are observing the monitored nodes.

The remaining chapter is organised as follows. Section 5.2 presents the explanation about the assumptions used in the detection scheme. This is followed by discussion about the data packet acknowledgement technique used in the simulation in Section 5.3. Section 5.4 introduces the proposed free-riding nodes detection scheme. Types of node action are explained in Section 5.5. Section 5.6 describes classification of observation used in the scheme. Section 5.7 gives a scheme walk-through related to the observation process. Section 5.8 explains the suspicious checking used in the scheme. Section 5.9 presents the process to make the final decision on the detection. Finally, the summary of the chapter is given in Section 5.10.

5.2 Assumptions

In this section, we discuss some of the assumptions used in the proposed detection scheme.

5.2.1 Behaviour Observability

In order to classify a node as normal or misbehaving, its behaviour has to be observable by other nodes. To ensure observability, we rely on two properties as explained below.

- Links are bidirectional. In such a condition, any node will be able to monitor traffic from nodes in the neighbourhood through promiscuous mode.
- Routing information in the header is unencrypted. This information is used as a basis to verify which nodes participate in forwarding packets.

5.2.2 Decentralised System

We assume the network is an open MANET. It may consist of mobile nodes that belong to many different organizations which may have different objectives and priorities. There is no trusted centralised server or prior trust relationship with any of the participating nodes. In this kind of environment, the nodes are free to choose whether to cooperatively become well-behaving nodes or decide to become misbehaving nodes.

5.2.3 Identity

The effectiveness of any detection mechanism in decentralised systems such as MANETs is largely dependent on the reliability of the identities used by the nodes in the network. When a node joins a network, it is required that the node has an identity which is persistent, unique and distinct. The requirement to be persistent means that a node cannot easily change its identity. This property is desirable for the detection systems to gather the behaviour history of a node. Secondly, unique identity is needed to ensure that behaviour observed was indeed that of the node observed. An identity is unique if no other node can use it and thus impersonate another node. Lastly, the requirement of distinct identities is the target of so-called Sybil attack analysed by Douceur [69], where nodes generate several identities for themselves to be used at the same time.

5.2.4 Preserve Resources

The misbehaving nodes that the scheme aims to detect are free-riding nodes which do not forward any packets for other nodes by not participating in the route discovery process. Free-riding nodes ignore any data and control packets that do not give benefit to them. The main objective of being free-riding nodes is to preserve their scarce resources (battery power, bandwidth etc) for their own use so that they can stay longer in the network. Free-riding nodes are different from malicious nodes in which the latter are not primarily concerned with power saving but only in attacking the network. Nevertheless, free-riding nodes degrade the performance of a network as already explained.

5.3 Passive Acknowledgement (PACK)

During packet forwarding, every intermediate node is responsible for confirming that the data packet was received by the next hop. There are three ways to get this confirmation and they have been briefly explained in Chapter 2. In our experiment, we implement passive acknowledgement. Passive acknowledgement (PACK) means that instead of waiting for an explicit acknowledgement for each data packet by the next-hop node on the route, a node assumes the correct reception of the packet when it overhears the next-hop node forwarding the packet. PACK can be used for Route Maintenance when originating or forwarding a data packet along any hop other than the last hop. PACK cannot be used with the last hop since it will never retransmit a packet destined for itself. PACK needs two conditions to be applied:

- The nodes have their network interfaces in promiscuous mode.
- The network links operate bi-directionally.

Due to the bi-directionality of the link-layer, a node is able to find out whether the next node forwards its data packet if both nodes are still in the range of one another. This is possible because in promiscuous mode, the node could also receives the packet when the next node forwards it to the next hop. When a node receives a data packet to be forwarded to a node other than last hop, the node will store a copy of the packet in a buffer. The packet is sent without requesting a link-layer acknowledgement (ACK). If the node does not overhear the packet forwarded, it assumes that the next hop either did not forward it or that it did forward the packet but was not overheard because the next-hop node moved out of range just after receiving the packet to be forwarded. With the PACK retransmission mechanism, the node waiting for the PACK resends the packet without a link-layer ACK request. After a certain number of trials, a link-layer ACK request must be used instead of PACK for all remaining attempts for that packet. If it does not get acknowledged, it emits a route error (RERR) claiming that the next node is unreachable.

When a node overhears a new data packet through promiscuous mode, it considers the packet as a PACK if the following checks succeed:

- The source address, destination address, protocol identification and fragment offset fields in the IP header of the two packets match.
- If either packet contains a DSR Source Route header, both packets must contain one, and the value in the Segments Left field (it indicates the number of hops remaining until the destination) in the DSR Source Route header of the new packet must be less than that in the stored packet.

The DSR draft [26] indicates the fields a node must check in order to consider the packet it receives as a PACK. By checking the four fields of the IP header, a node can identify a packet uniquely so that it can be sure that the packet it overheard is the retransmission of the one it forwarded. Next, the DSR draft requires that if both packets have a source route, then the segments left value in the overheard packet must be less than in the stored packet. The second requirement assures that the overheard packet is fresher than the stored one.

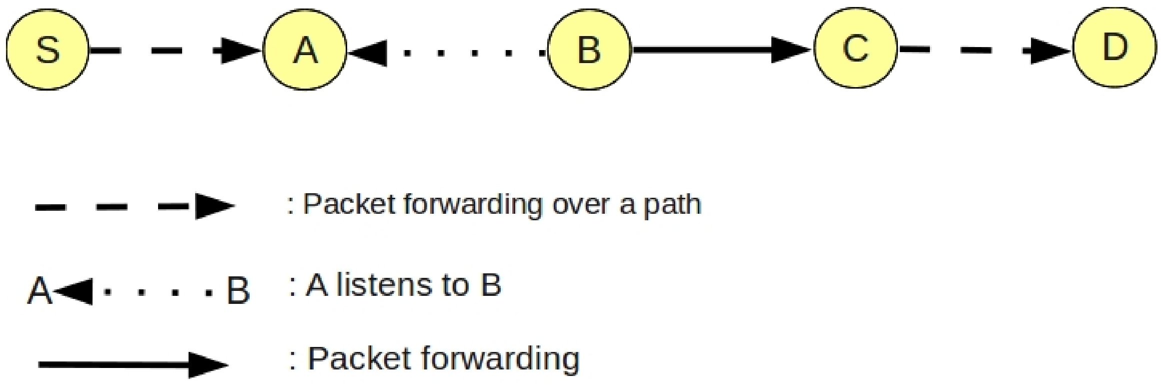
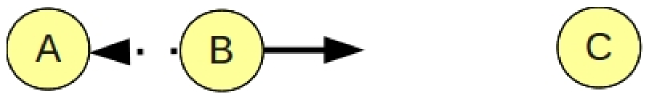


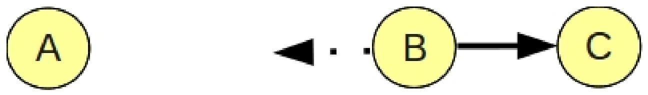
Figure 5.1: Passive Acknowledgement

Apart from its advantages, passive acknowledgement has several disadvantages which are caused by the peculiarities of mobile ad hoc networks. Passive acknowledgement may not work efficiently with the presence of ambiguous collision, receiver collision and limited transmission power. As shown in the scenario demonstrated in Figure 5.1, an ambiguous collision occurs at A while it is listening for B to forward a packet on. Node A does not know whether the collision is caused by its neighbouring nodes or by node B. In the receiver collision problem, node A can only tell whether B sends the packet to C but it cannot tell if C receives it. Node C might not receive the packet because of a collision. Another problem is that a node can limit its transmission power such that a signal is sufficiently strong to reach the previous node while weak enough not to reach the true recipient as shown in Figure 5.2a. In another scenario illustrated in Figure 5.2b, the next node receives the packet but the power of the transmission is too low for the previous node to overhear the packet.

Passive acknowledgement requires the nodes to operate in promiscuous mode. When a node is operating in promiscuous mode, the MAC layer does not filter any packet that it receives but delivers all these packets to the higher layer. This results in the node being able to receive all packets being transmitted in its neighbourhood. Promiscuous monitoring also has significant limitations. These are several



(a) Node A overhears node B forwards packet but the packet does not reach node C



(b) Node C receives the packet but node A cannot overhear node B forwards packet

Figure 5.2: Limited Transmission Power

network adapters that do not support promiscuous monitoring since the corresponding drivers at the MAC layer might not allow the filtering to be turned off. The promiscuous mode of operation also increases the power consumption because it requires the nodes to monitor transmission not destined for them. According to [70], the energy consumption due to receiving packets is almost the same as the energy for transmitting. In other words, a node operates in promiscuous mode uses more energy that a node that operates in non-promiscuous mode.

As mentioned before, passive acknowledgement cannot be used if the next-hop node is the final destination. A node will not retransmit packet destined to it. In that case, we make use of link-layer ACK for the transmitting node to ensure the destination node receives the packet and still within range.

In our simulation, we use the simple passive acknowledgement not only for the indication of correct reception at the next hop, but also for failure detection when the next-hop node fails to forward packets. Every time a node detects the next-hop node forward a data packet, it will update a success counter (numForward). The node also has another counter (numDrop) to store the number of the failure forwarding. These counters will be used later by the node to make a decision in the detection scheme.

5.4 Proposed Free-Riding Nodes Detection Scheme

In our proposal, each monitoring node operates in promiscuous mode and would monitor both data and control packets that are sent within its receiving range. Each monitoring node will keep a record for each of its neighbouring nodes. In our simulation, each one of the nodes in the network is a monitoring node regardless of its preset behaviour (well-behaving or free-riding). In the INETMANET Framework [61], there is already a specific table to store the information about neighbouring nodes. We add extra fields to the table as the following:

1. numForward
2. numDrop
3. last_action
4. last_request
5. passNum
6. failNum
7. status

numForward is the number of data packets forwarded while numDrop is the number of data packets failed to forward by the neighbouring node. last_action is the time the neighbouring node is last seen contributing or providing services to the network. last_request on the other hand is the time recorded the neighbouring node is last seen utilizing or requesting for services from the network. These two fields would be updated for every action observed due to the promiscuous monitoring mode.

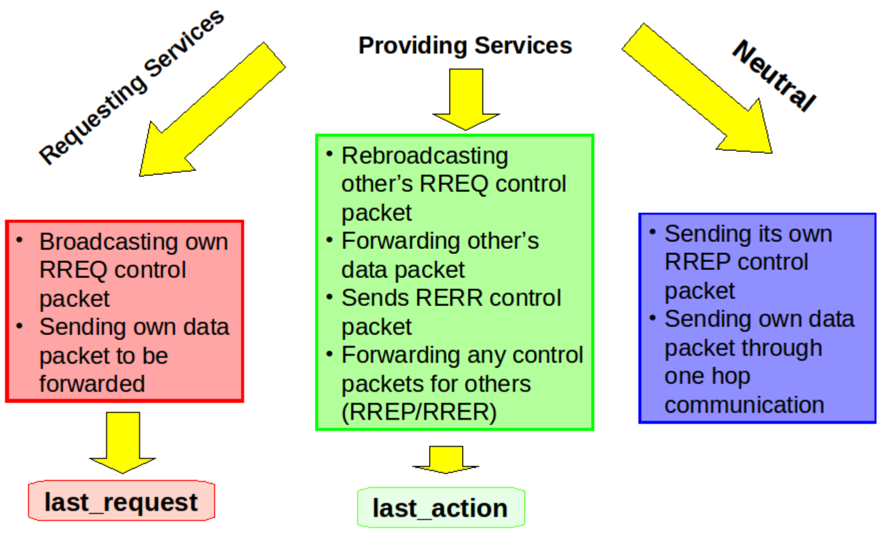

Requesting Services
Providing Services
Neutral
• Broadcasting own RREQ control packet
• Sending own data packet to be forwarded
• Rebroadcasting other's RREQ control packet
• Forwarding other's data packet
• Sends RERR control packet
• Forwarding any control packets for others (RREP/RRER)
• Sending its own RREP control packet
• Sending own data packet through one hop communication
last_request
last_action

Figure 5.3: Types of Actions

passNum is the number of times the monitored node passed the suspicious test which will be elaborated on later. failNum on the other hand is the number of times the monitored node fail the suspicious test. Both passNum and failNum will be set to zero in the beginning of the simulation. Finally, status is the current behaviour of the neighbouring node detected by the monitoring node. The initial value for field status for any neighbouring node is initially set to zero for unknown. The status field could later be changed to suspicious or behaved as we will explain later.

5.5 Categories of Node's Actions

We have identified types of actions that are considered as utilizing, contributing or neither of those. These are depicted in Figure 5.3.

5.5.1 Utilizing

In this category, the node uses the services provided by other nodes. There are two actions under this category. First is when the node requests its neighbouring node to forward its own data packet to the next node through multi-hop communication. Second is when the node broadcasts its RREQ control packet to the network when searching for a route to the destination node.

5.5.2 Contributing

Whenever a node give services to another node, its action is considered as contributing to the network. By doing so, the node has to use some part of its resource to provide services. In a MANET, for the network layer, services can be either routing discovery or packet forwarding. In order to be able to send a data packet on behalf of others, the node has be a part of the routing. One important thing regarding actions in this category is that a node can only provide services if it is requested by other node. Nodes located at the edge of the network may have little chance to provide services under this group.

5.5.3 Neutral

In this case, the action a node performed can neither be considered contributing to nor utilizing the network. Not utilizing does not always mean the mentioned node does not request other nodes to forward its packet. As we will see later, in some scenarios, the mentioned node may do so. However, that action is just to follow the routing protocol and not for its own benefit.

There are two events that fall into this categories: sending RREP control packets and sending data packets directly to the destination. When a destination node receives a RREQ control packet that is targeted to it, it will reply with a RREP control packet to complete the route information back to the source node. In this case, the event of route searching is initiated by the source node and not by the destination. So, the destination node should not be considered as using the network even though the RREP control is originated from it and may have to be forwarded by other nodes to reach the source node. In the second scenario, the source node sends its data packet directly to the destination node without having to be forwarded by any intermediate node. The source node does not require help for any of its neighbouring node and this action should be categorized as neutral.

5.6 Classification of Observations

We implement two types of observation: personal experience and direct observation. These type of observations fall under first-hand observation and have been briefly discussed in a previous chapter (chapter 3).

5.6.1 Personal Experience

Personal experience of a monitoring node refers to the information it gathers through one-to-one interaction with its neighbouring nodes. For every packet received, the monitoring node will check under which group (utilizing, contributing or neutral) the packet should be in and update the necessary field. However, if the monitoring node receives a data forwarding request it will also check if the time difference between last_request and last_action of the requestor exceed a threshold. If it does, the monitoring node receiving the data forwarding request will run a special test before deciding whether the monitored node should be labelled as free-riding or not. The flowchart in Figure 5.4 shows the process flow for personal experience observation.

5.6.2 Direct Observation

Direct observation refers to the information a node gathers by observing the interactions among its neighbouring nodes without being involved in the interaction. Similar to the personal experience steps, for every packet received for the neighbouring nodes, the monitoring node will also decide under which group (utilizing,

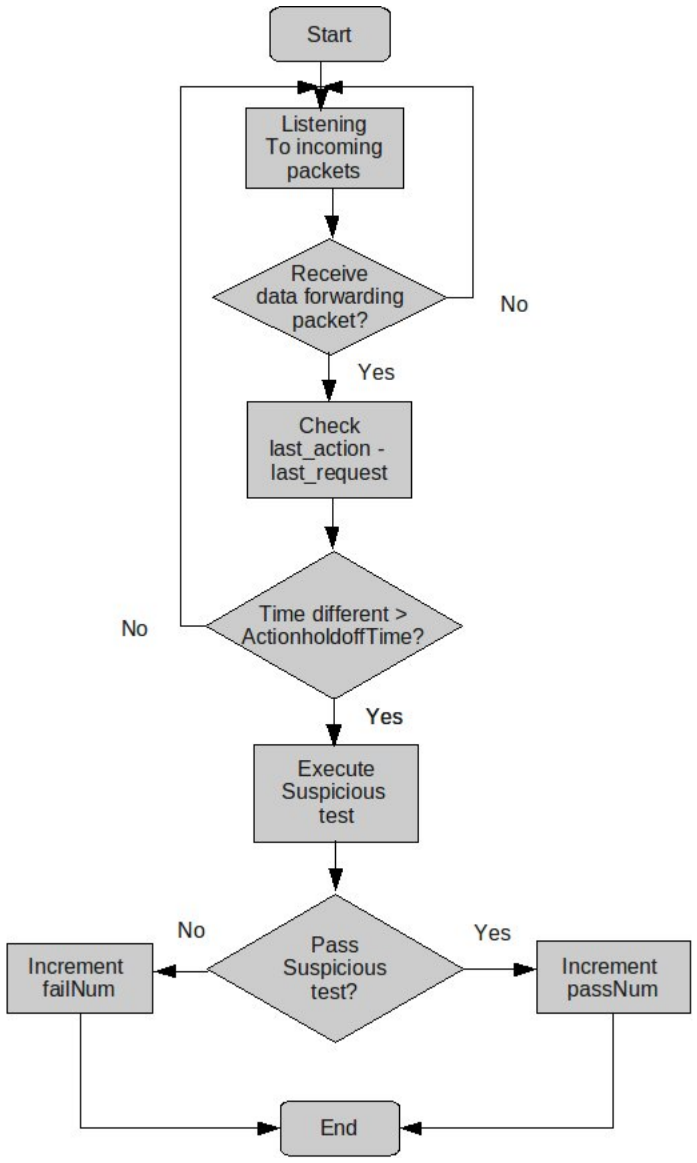

Start
Listening To incoming packets
Receive data forwarding packet?
No
Yes
Check last_action - last_request
Time different > ActionholdoffTime?
No
Yes
Execute Suspicious test
Pass Suspicious test?
No
Yes
Increment failNum
Increment passNum
End

Figure 5.4: Personal Experience Flowchart

contributing or neutral) the packet should be in and update the necessary field.

Whenever a monitoring node hears its neighbouring node request other nodes to forward a data packet, it will also check if the time difference between last_request and last_action of the requestor exceeds a predefined threshold. If it does, the monitoring node will wait for a period of time to give a chance for the requestor to contribute to the network before deciding whether the monitored node should be labelled as free-riding or not. The flowchart in Figure 5.5 shows the process flow for direct observation.

5.7 Observation Type Walk-Through

Figure 5.6 illustrates the difference between personal experience and direct observation. It also show examples of events for which the fields last_request and last_update will be updated. In the figure, there are five nodes with a circle surrounding each node. The circles with a node in every centre of them indicate the receiving range that the node has.

In Figure 5.6a, node 2 wants to send a data packet to node 8 through the intermediate node (node 1). When node 1 receives the data forwarding request, it will check if the packet is for itself. Since the packet is to be forwarded, it updates its field of last_request for node 2. In this example, node 1 gathers information about node 2 through personal experience because node 1 is directly involved in the data forwarding.

Since nodes 4 and 5 are in promiscuous mode and also within the range, they can also overhear the packet sent from node 2. They will also check if the packet is to be forwarded or is just for one hop only. Since that packet applies for the former case, they update their field of last_request for node 2. In this case, nodes 4 and 5 update their information about node 2 through direct observation because they are

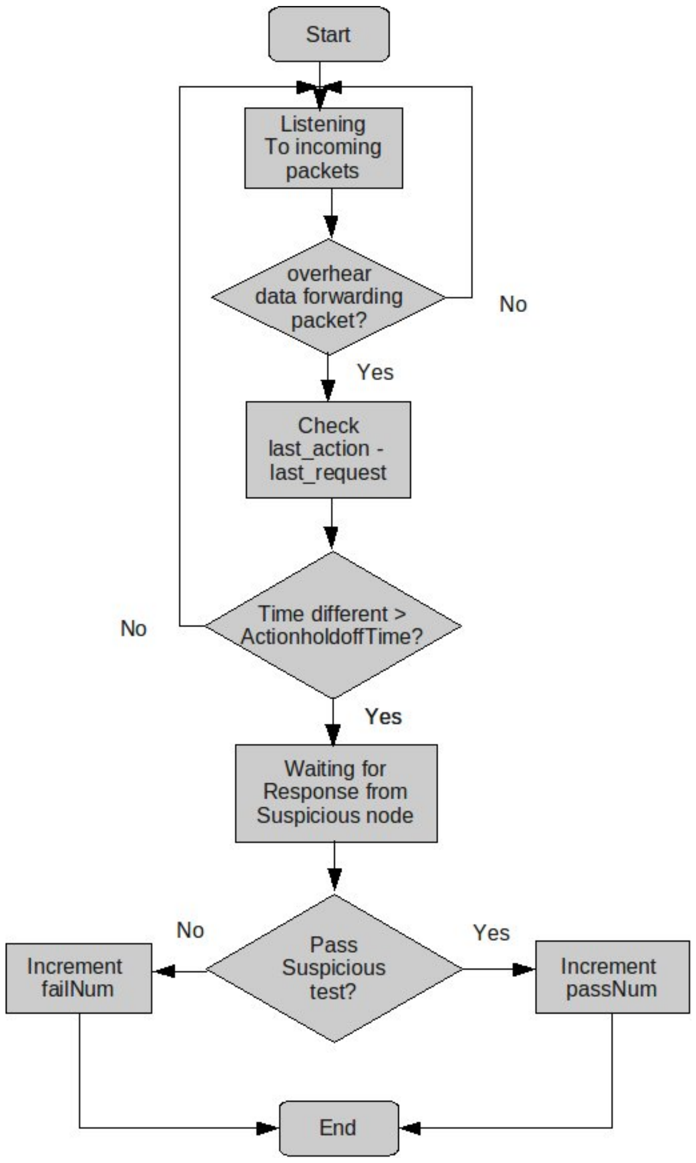

Start
Listening To incoming packets
overhear data forwarding packet?
No
Yes
Check last_action - last_request
Time different > ActionholdoffTime?
No
Yes
Waiting for Response from Suspicious node
Pass Suspicious test?
No
Yes
Increment failNum
Increment passNum
End

Figure 5.5: Direct Observation Flowchart

not directly involved in the activity.

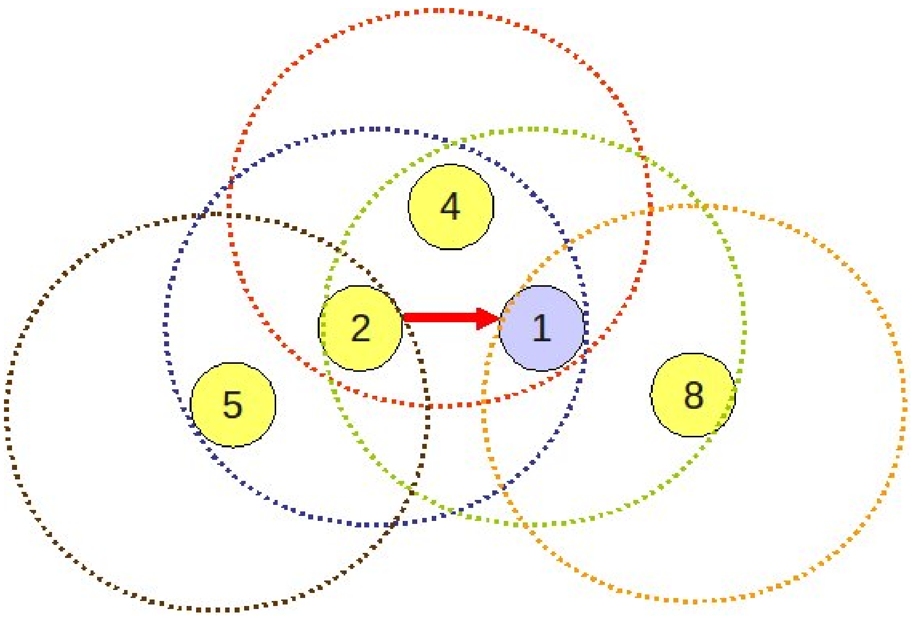
Later, in Figure 5.6b, the intermediate node (node 1) forwards the data packet it received to node 8 which is the next hop and also the destination for the packet. Node 8 then checks if the packet has come from more than one hop. This checking can be easily done by comparing the address of the previous node and the source node. In this case, the two addresses are different and node 8 will update last_action of node 1 for forwarding other's node data packet. This action can also be overheard by the neighbouring nodes (nodes 2 and 4) which will also update the field of last_action for node 1. In addition, since the node 2 is the previous hop for the data packet, it will also delete the copy of the packet in its buffer in accordance to DSR routing protocol and increment the number of forwarded data (numForward) for node 1.

In all of these cases, all of the respective monitoring nodes (1, 4 and 5) will always check the time difference of the requestor every time they encounter a data forwarding request. If the time difference is still within a threshold as in Figure 5.7, the status field for the node is set to behaved and no suspicious test is necessary. We name this threshold as ActionHoldoffTime.

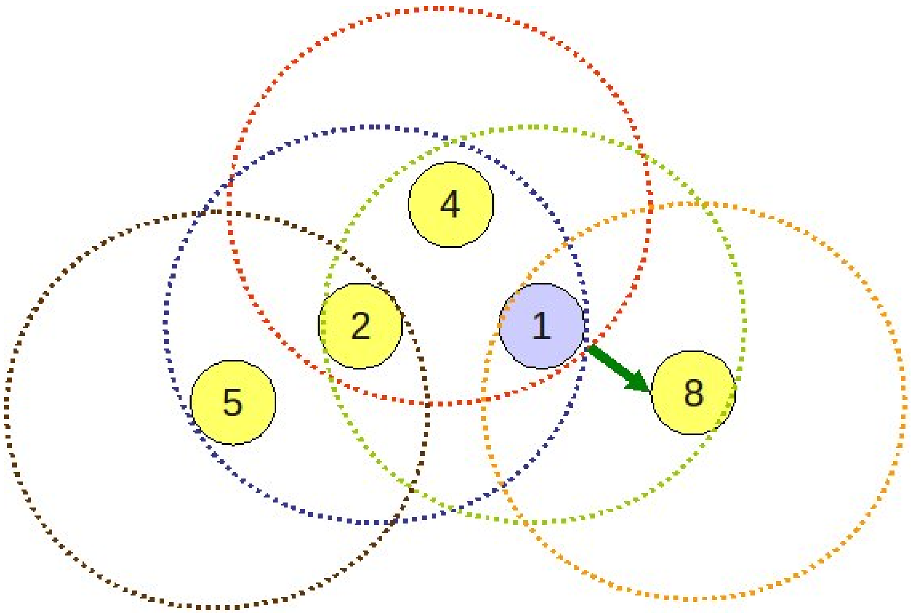
5.8 Suspicious Testing

If the time difference exceeds the threshold, the status field for the node will be set to suspicious. At this stage, further testing should be conducted as this suspicious node might be wrongly monitored as misbehaving due to unintentional packet drops or the special scenarios as explained above.

In this testing, a fake RREQ packet will be injected into the network as depicted in Figure 5.8. To minimized traffic flooding in the network, only the monitoring node that receives data forwarding request from the suspicious node should conduct this testing. In addition, this fake RREQ packet should only be allowed to pass through



(a) Last_Request Update



(b) Last_Action Update

Figure 5.6: Observation Walk-Through

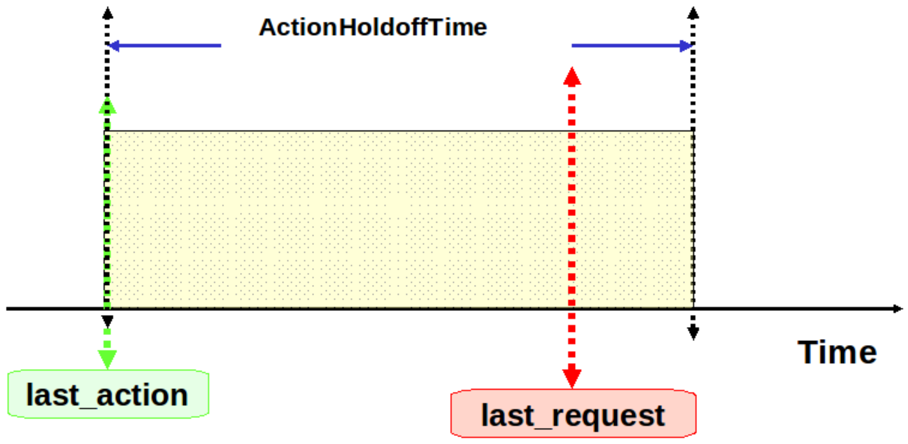
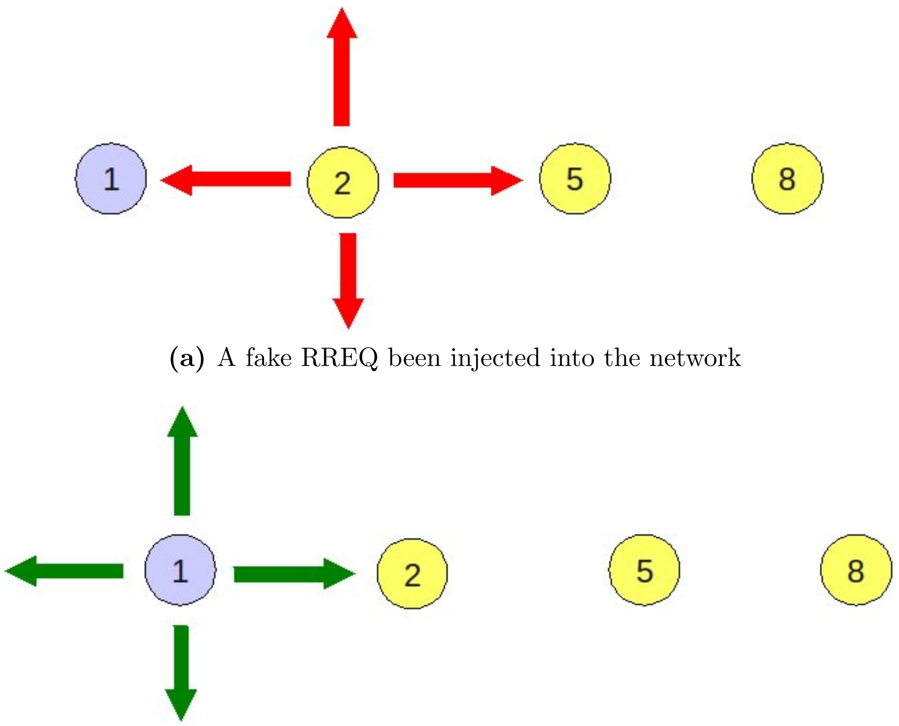


Figure 5.7: ActionHoldoffTime



(a) A fake RREQ been injected into the network

(b) Good node rebroadcast RREQ

Figure 5.8: Suspicious Testing

one hop (TTL=1). All other monitoring nodes in the neighbourhood that detect this potential misbehaviour would wait for the suspicious node to rebroadcast the fake RREQ packet within a certain time period. If the suspicious node responds to the RREQ packet, the status field of the node is set to behaved and the time of its last_action will be updated. In addition, the value of passNum for the suspicious node will be incremented. If after the timeout the node discards the packet and does not respond, the monitoring nodes will finally increment value of failNum for that suspicious node.

When a monitoring node broadcasts the fake RREQ packet, all of the monitoring nodes in the neighbourhood see the packet as a normal control packet. The neighbouring nodes will regard this action as utilizing the network and update the variable last_request accordingly. This will not effect the node running the testing as it will not be accused as a free-riding node as long as the time difference for the node is still within the threshold. Even if the time difference exceeds the threshold, the node running the suspicious testing will have a chance to pass the suspicious test conducted by the other monitoring node. In our scheme, each monitoring node will only consider its own personal discovery and will not be share this observation with other nodes. This eliminates most of the data dissemination complexity and avoids any false accusation and false praise attacks.

5.8.1 Fake RREQ Packet

In the testing procedure explained above, a fake RREQ packet is used for the following two purposes:

1. To test if the node requesting to forward its data packet is free-riding node.
2. To give opportunity for well-behaving nodes to contribute to the network so

that they will not be labelled as free-riding nodes.

A fake RREQ is a normal RREQ control packet which has a destination address of a nonexist node. It has to be for a nonexistant address to ensure that no node will respond with RREP control packet and then jeopardize the suspicious testing. For example, in the DSR routing protocol, a node does not have to rebroadcast RREQ if it has already heard other node replies with RREP.

There are many methods to choose a fake address. In our experiment, we simply add by one the third octet of the targeted IP address. For example, if the IP address of the node to be tested is 192.168.1.1, the fake address would be 192.168.2.1. In this way, it is easier for the node doing the testing to trace which node is being tested and determine the RREQ control packet expected.

5.9 Decision Making

In this section, we explain the process the proposed detection scheme uses to make a decision. The basic decision is a binary decision, on which of the neighbouring nodes are free-riding and which are not. It is worth understanding that in our simulation, each node makes its decision based on its own observation without any influence from other nodes. The decision is based on the following parameters:

- numForward - number of forwarded data packets by the neighbouring nodes.
- numDrop - number of dropped data packets by the neighbouring nodes.
- passNum - number of successful suspicious test.

- failNum - number of failed suspicious test.

In order to be labelled as a well-behaving node, the monitored node has to pass the following requirement

$$numDrop + failNum = 0 \tag{5.1}$$

In the extreme detection evaluation, the monitored nodes should never be observed to drop any data packet or fail the suspicious test. If they do, the monitoring nodes would labelled them as free-riding nodes.

Later in the experiment, we introduce a forgiveness threshold when detecting free-riding nodes. In wireless communication, packet drops due to collision, fading, etc., are not uncommon. To limit the number of well-behaving nodes being accused as free-riding due to the unintentional behaviour, the scheme allows the nodes to fail on a certain percentage. In other words, equation 5.1 could be non-zero. In our experiment, we set this forgiveness threshold to 10%. So in this case, to be labelled as a well-behaving node, the monitored node should pass the following requirement:

$$\frac{numForward + passNum}{numForward + passNum + numDrop + failNum} > 0.9 \tag{5.2}$$

5.10 Summary

In this chapter, the proposed scheme to detect free-riding nodes in a MANET has been discussed in detail. First, the assumptions of the scheme are briefly explained. Passive acknowledgement which is the receipt confirmation technique used in the scheme is discussed. In the detection scheme, each node works independently and monitors the actions of its neighbouring nodes by analysing the packets it receives through promiscuous mode. Those actions are grouped into three categories and are discussed along with some examples. Two types of observation are used in the scheme and they are compared in detail in the walk-through section. To handle certain types of nodes, suspicious checking that injects a fake RREQ control packet to the network is implemented in the scheme. The checking process and the the fake RREQ control packet used are described. Lastly, the decision making process in the detection scheme is discussed. The performance evaluation of the scheme along with the analysis will be discussed in the next chapter.

Chapter 6

Performance Analysis

6.1 Introduction

This chapter will discuss the performance of the proposed free-riding detection scheme in a decentralised and open system such as a MANET. In such an environment, the nodes in the network may belong to different authorities. Each individual node is free to make its own decision whether to be cooperative with other nodes or not. Without prior agreement between the nodes, each node also has to conduct its own observation and judgement when performing the detection scheme without relying on other nodes. To assess the ability of the scheme to detect free-riding nodes, performance analysis metrics are evaluated and compared under a range of conditions.

6.2 Performance Evaluation

6.2.1 Performance Analysis Metrics

When evaluating our proposed scheme, we consider the following performance metrics

6.2.1.1 True Detection Ratio:

True Detection Ratio (TDR) or true positives represents the average of true detection computed as follows:

$$TDR = \sum_{i=1, m_i \neq 0}^{n} \frac{td_i}{m_i} * 100 \tag{6.1}$$

td_i: number of free-riding nodes monitored and detected by node i

m_i: the number of free-riding nodes monitored by node i

n : number of nodes

6.2.1.2 False Detection Ratio:

False Detection Ratio (FDR), or false positives, represents the average ratio of false detection using the following formula:

$$FDR = \sum_{i=1, m'_i \neq 0}^{n} \frac{fd_i}{m'_i} * 100 \tag{6.2}$$

fd_i : number of well-behaving nodes monitored and wrongly detected by node i

m'_i : the number of well-behaving nodes monitored by node i

n : number of nodes

Obviously, a good detection scheme is where the TDR is high and FDR is low.

6.2.1.3 Routing Overhead (RO)

Routing overhead is the ratio of the total number of routing packets transmitted per data packets delivered at the destination during the simulation using the following formula:

$$RO = \sum_{i=1, d_i \neq 0}^{n} \frac{c_i * S_r}{d_i * S_d} \tag{6.3}$$

c : total number of control packets send by node i

S_r *: size of control packet*

d_i*: total number of data packets originating from node i*

S_d *: size of data packet*

In the context of this experiment, routing overhead is the average number of routing packets required to deliver a single data packet. Each hop wire transmission of a routing packet is counted as one routing packet. Routing overhead is an important metric since it provides an indication of the extra bandwidth consumed by overhead to deliver data traffic and has direct impact on the studied scheme efficiency. The lower average routing overhead, the better the scheme is.

6.2.2 Simulation Environment

OMNET++ network simulator has been used to simulate the proposed scheme in detecting free-riding nodes in a MANET. The simulated network contained 20 mobile nodes randomly distributed in a playground area size of 500 m by 500 m. In the simulation, each node is both a source and destination for other nodes. For each data packet, the source node will randomly select the destination for the packet. This is the worst case scenario because the source node has to find the route to almost all other nodes in the network. Furthermore, all nodes in the network are transmitting and receiving data packets for the whole simulation time. The data packet size used is 512 bytes. Traffic sources of constant bit rate (CBR) are used where the source node sends data packets with a frequency of 5 packets per second. The underlying MAC protocol defined by IEEE 802.11 is used with a channel data rate of 54 Mbps.

Table 6.1: Simulation parameters for Detection Scheme

parameter	Value
Simulation Time	100 seconds
Number of Nodes	20
Packet Size	512 bytes
Network Area	500m x 500m
Mobility Model	Random WayPoint
Transmission Range	250 metres
Routing Protocol	Dynamic Source Routing (DSR)
Data Rate	5 packets/second

The wireless transmission range of each node is 250 metres. When the node's speed is set to zero, a null mobility model is chosen. Otherwise, a random way-point mobility model is assumed with a pause time of 0 seconds.

In order to evaluate the detection capability of our scheme in a relatively short period of time, we only set the simulation time to 100 seconds. Table 6.1 summarizes the parameters used in our simulation.

We simulated our scheme using four configurations of free-riding nodes in the network as listed below:

1. no free-riding node.
2. 5 free-riding nodes.
3. 10 free-riding nodes.
4. 15 free-riding nodes.

We do not use a network where all 20 nodes are behaving as free-ridings because in such an environment no node will offer a packet forwarding service. Furthermore, there is no great benefit conducting free-riding nodes detection when every single node is selfish (network is bad).

6.3 Simulation Results

The main objective of the simulation is to measure the ability of the scheme to detect free-riding nodes in a MANET under ranges of conditions. Performance metrics must be measured against some parameters that describe the characteristic behaviour of a mobile ad-hoc network and can be varied in a controlled manner. The parameters chosen are:

- Size of ActionHoldoffTime
- Mobility (speed of the nodes)
- Number of free-riding nodes in the network

In this section, simulation results to evaluate the performance of the proposed free-riding detection scheme will be presented.

6.3.1 Impact of varying ActionHoldoffTime and Mobility Speed

In our first result, we evaluate the TDR and FDR of the studied detection scheme against the mobility speed of the nodes and also the size of ActionHoldoffTime. For each configuration, we evaluate the scheme by varying the node's speed from 0 m/s

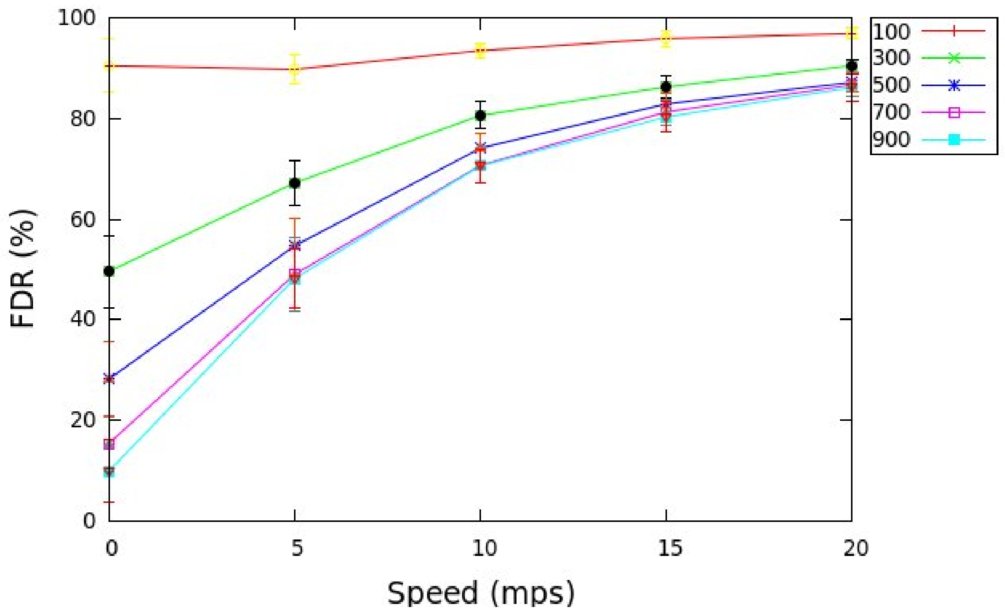


Figure 6.1: FDR with Different Node Mobility Speed

to 20 m/s and the window size of ActionHoldoffTime from 100 ms to 900 ms. The results presented for each data set are the average of 40 simulation runs, each with different seeds.

6.3.1.1 True Detection Ratio

From the simulation, we found that the results of the TDR are always 100% regardless of the nodes' mobility speed and the value set for ActionHoldoffTime. That means, using our scheme, each node is able to detect all of its neighbouring nodes that are behaving as free-riding nodes.

6.3.1.2 False Detection Ratio

However, the finding results of FDR give different conclusion. As depicted in Figure 6.1, the results of FDR never reach zero in any set of parameters. That means, some

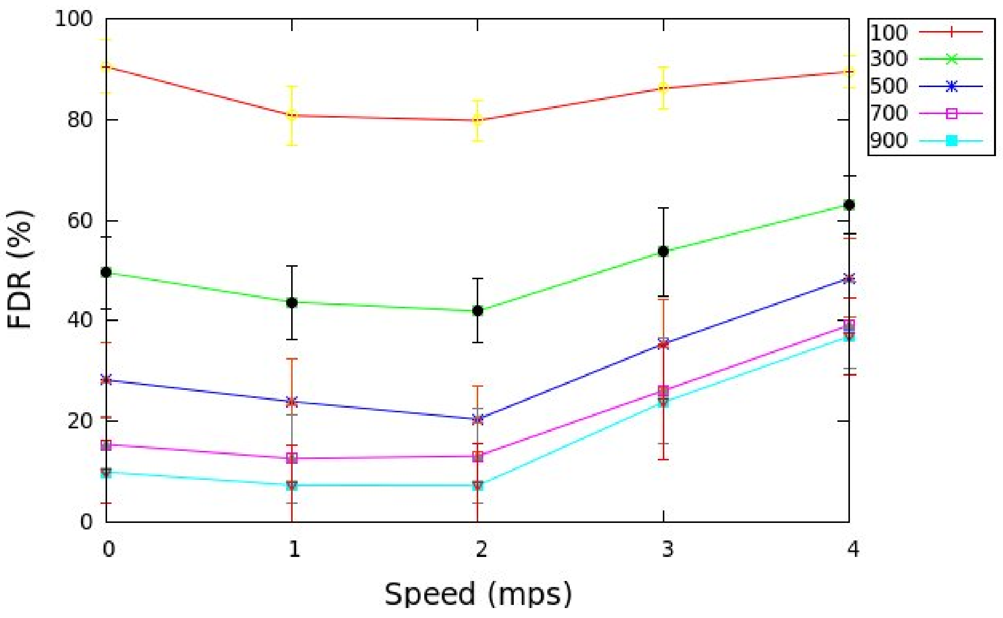


Figure 6.2: FDR with Small Node Mobility

well-behaving nodes have been wrongly detected as free-riding nodes. From that figure, we found that the results of FDR are affected by both the size of ActionHoldoffTime and the mobility speed. The lower the value set for ActionHoldoffTime, the higher percentage of well-behaving nodes are wrongly detected as free-riding nodes.

We can also see that the results of FDR also increases as the speed of the nodes increases. As the speed of the nodes increases, the probability of link failure increases and hence the number of packet drops also increases. With the number of packet drops increased, more well-behaving nodes would be detected as free-riding nodes.

The effect of the mobility speed on the final results of FDR is greater when the value of ActionHoldTime is higher. For example, when the mobility speed is increased from 5 m/s to 10 m/s, the results of FDR increase by 20% when Action-HoldoffTime is 300 ms compared to 44% when ActionHoldoffTime is 700 ms.

Next, we ran the same experiments but with mobility speed between stationary and 4 mps which is considered relatively slow mobility. In the previous graph, it looks like value of FDR is increasing constantly with the node mobility speed. However, Figure 6.2 shows that the value of FDR actually decreases when the nodes move slowly. The FDR reaches the minimum when the mobility speed is 2 m/s and then it starts to increase as we increase the nodes' mobility speed.

The value of FDR is related to the packet collision and the link break. If the density of nodes is very high (many nodes in the coverage area), the probability of a collision is very high. When all nodes are not moving, the density of nodes will be static which may lead to consistent packet collisions. As the nodes start moving around in the playground area, the density of nodes in the network is changing. However, if the mobility of the nodes is too high, the number of packets loss due to the link breaks also increase. Packets loss due to these two factors leads to the wrong observation by the monitoring nodes. From the simulations, based on the parameters set, at the node mobility speed of 2 m/s, the packet collision and the number of link break is at the minimum.

6.3.2 Impact of number of free-riding nodes in the network

In this section, we want to investigate how the detection copes with a varying numbers of free-riding node in the network. We fix the mobility speed to 2 mps but vary the number of free-riding nodes in the network from 0, 5, 10 and 15. We also vary the size of ActionHoldoffTime from 100 ms to 900 ms to see its effect to the final results. The results presented for each data set are the average of 40 simulation runs, each with different seeds.

6.3.2.1 True Detection Ratio

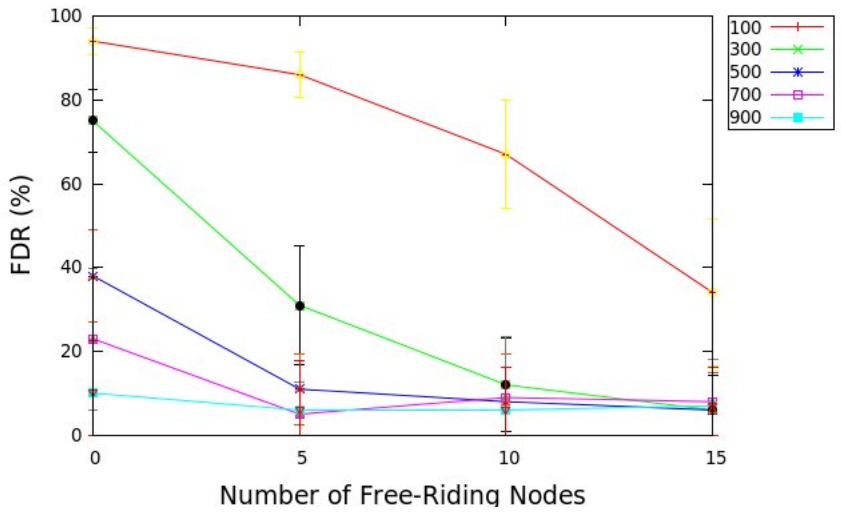
In all set of the parameters used (varying number of free-riding nodes) in the simulation, the results of TDR remain 100%. Using the scheme, each node is able to detect all possible free-riding nodes in its neighbourhood regardless of their number in the network.

6.3.2.2 False Detection Ratio

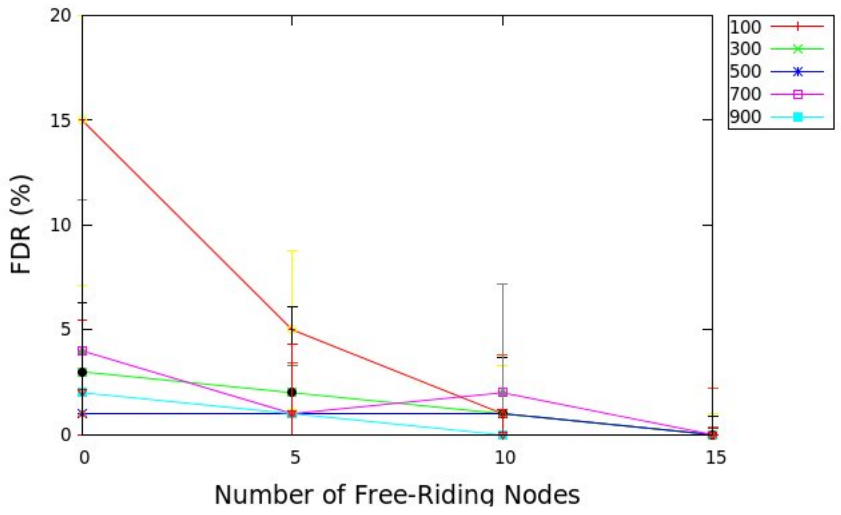
From Figure 6.3a, we found that the scheme has lower FDR when the number of free-riding nodes in the network are higher. That means the scheme is performing better when there are a greater number of free-riding nodes in the network.

When all of nodes are well-behaving, all of them actively rebroadcast RREQ control packets and become part of the routes. As the number of well-behaving nodes increases, the amount of traffic packets in the network also increases. In this scenario, the probability of having a packet dropped due to network congestion or collision is higher compared to when there are many free-riding nodes in the network. The dropped packets would lead the source node to retransmit or even initiate a fresh route discovery to find a new path to the destination. All of these contribute to the increased amount of traffic in the network.

The effect is more significant when the value for ActionHoldoffTime is lower. The lower value set to ActionHoldoffTime reduces the time frame for the monitoring nodes to check on their neighbouring nodes and thus increases the number of suspicious checks to be conducted. Suspicious checking injects RREQ control packet to the network which in turn contribute to packet dropping due to network congestion and packet collision. With the greater possibility of dropping packet unintentionally, more well-behaving nodes would be mistakenly detected as free-riding nodes by the



(a) FDR without 10% forgiveness



(b) FDR with 10% forgiveness

Figure 6.3: FDR with Different Number of Free-Riding Nodes

detection scheme.

In Figure 6.3b, we use 10% forgiveness when conducting the detection. That means the nodes being monitored are allowed to drop 10% of the forwarding requests before they are labelled as free-riding nodes. The direct affect of this forgiveness is the decrease results of the FDR. Less number of well-behaving nodes are labelled as free-riding nodes. However, the results are also influenced by the size of the ActionHoldoffTime and the number of free-riding nodes in the network. FDR with 10% forgiveness is highest when there is no free-riding nodes and the size of ActionHoldoffTime is 100 ms.

6.3.3 Average routing overhead

In this section, we measure the average routing overhead that occurred due to the implementation of the detection scheme. One reason for the increase of routing overhead is because of fake RREQ control packets that are broadcast by the monitoring nodes during suspicious checking. The other contribution for this increase is the rebroadcast RREQ control packets by the neighbouring nodes. The neighbouring nodes 'could' not differentiate the normal RREQ control packet from the fake RREQ used in the suspicious check. When they receive any RREQ control packet that is not destined for them, they rebroadcast it back if the TTL of the packet is not zero.

In this experiment, we vary the nodes' mobility speed from 0 to 4 m/s and the size of ActionHoldoffTime from 100 ms to 900 ms. We calculate the routing overhead and take the average of the routing overhead when the number of free-riding nodes are 0, 5, 10 and 15. Finally, we compare these results when no suspicious checking is used. The results presented for each data set are the average of 40 simulation runs,

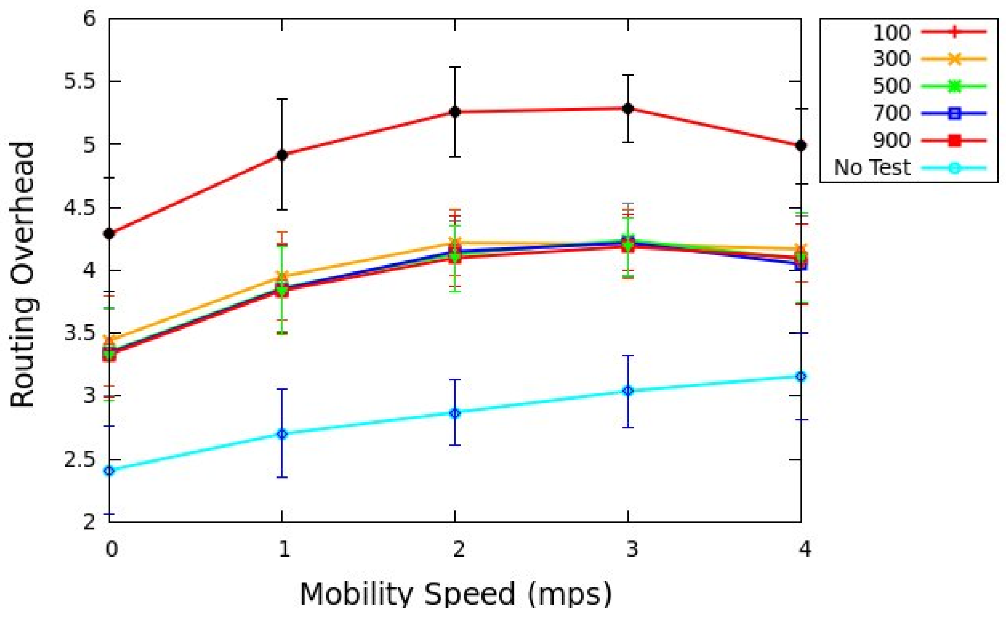


Figure 6.4: Routing Overhead Comparison with Respect to Speed

each with different seeds.

From figure 6.4, we can see that when suspicious testing is not used, the average routing overhead increases almost consistently with respect to the mobility speed. When suspicious testing is used, in every set of scenarios, the average routing overhead increases dramatically compared to when the testing is not used. The increase is more obvious when the size of ActionHoldoffTime is 100 ms. When the size of ActionHoldoffTime is between 300 ms and 900 ms, there is no major difference in the average routing overhead.

6.3.4 Effect of suspicious checking

In this section, we discuss the effect of suspicious checking on the final results of the detection scheme. In the previous chapter, we describe some scenarios where suspicious checking is required. Bear in mind that by conducting suspicious checking, the monitoring nodes have to inject fake packets and this activity consumes some energy

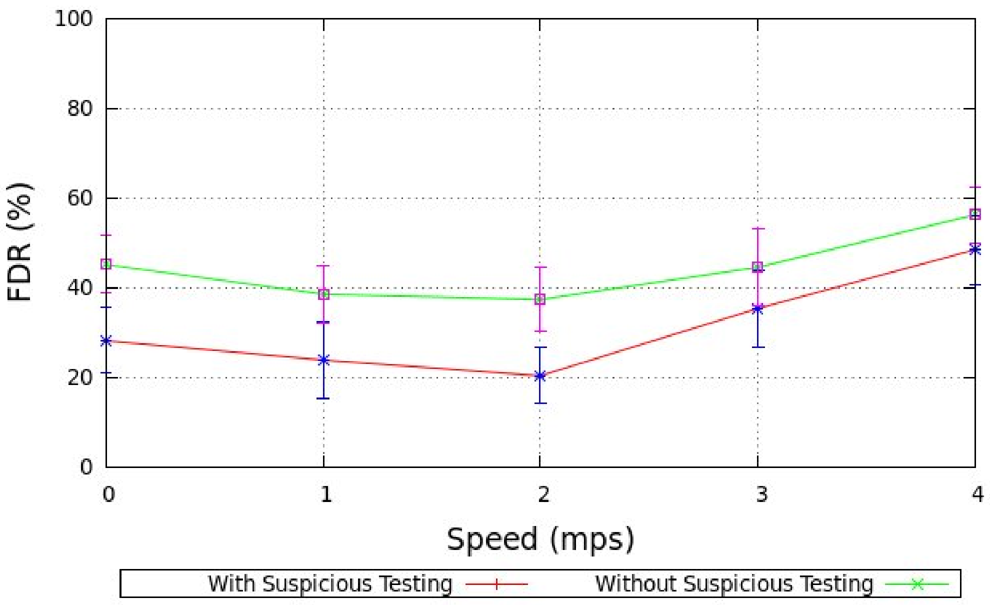


Figure 6.5: Effect of Suspicious Testing in Lower Speed

and increases the amount of traffic in the network. Based on previous experiment results, suspicious checking increases the average routing overhead. More control packets are needed to be sent into the network to deliver the same number of data packets.

We want to study using network simulations whether the suspicious checking is really needed and evaluate how it influences the performance of the detection scheme. In this experiment, we simulate a scenario where suspicious checking procedure is not used. Whenever a monitoring node detects that its neighbouring node fails to contribute to the network within the predefined time period, the monitoring node will simply increase the fail counter of the neighbouring node. For this set of simulations, we fix the size of ActionHoldoffTime to 500 ms and vary the mobility speed from 0 mps to 20 mps. The results presented for each data set are the average of 40 simulation runs, each with different seeds.

Figure 6.5 and 6.6 show the relationship between the mobility speed and False Detection Ratio (FDR) when using and without the use of suspicious checking. From

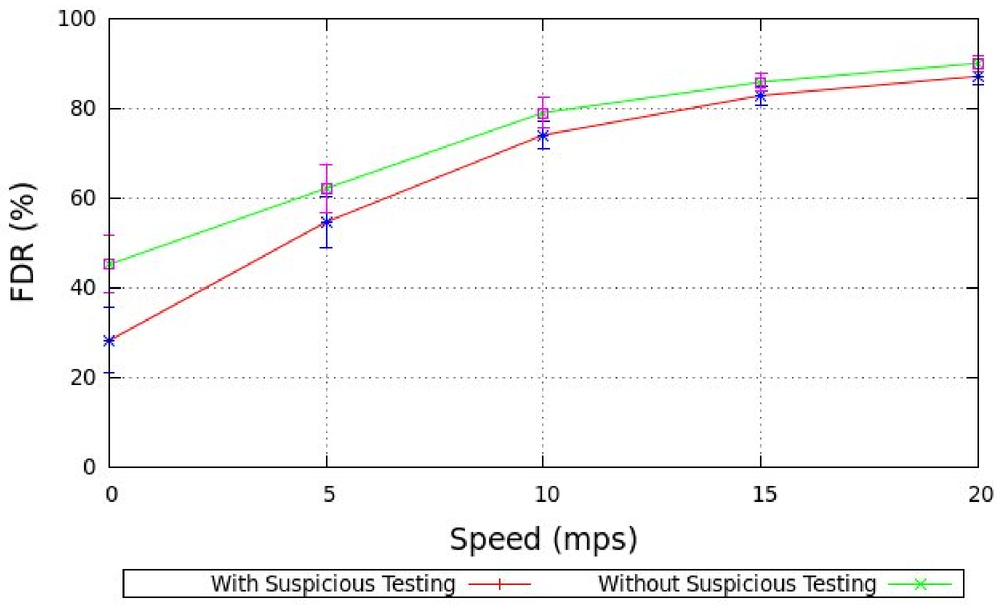


Figure 6.6: Effect of Suspicious in Larger Scale

the figure, we can see that the results of FDR are lower when suspicious testing is used at set of mobility speed. In other words, the number of well-behaving nodes wrongly detected as free-riding nodes are reduced when suspicious checking procedure is used. The procedure gives a more significant effect when the mobility speed is between 0 m/s to 2 m/s. The detection results are better when suspicious testing is used because it gives the well-behaving nodes a second chance to contribute to the network and avoid been accused of being free-riding nodes. When the nodes moves at a higher speed, the network topology changes more dynamically. There will be a higher possibility for the packets to be lost during the transmission. As a result, the possibility for those well-behaving nodes to pass the suspicious testing will be lower.

6.3.5 Partial Free-riding

In our previous experiments, we assume all of the free-riding nodes are consistently behaving like one. Free-riding nodes drop all control packets not destined for them and thus are not be requested to forward data packets for others. In this section, we evaluate our scheme in scenarios where the free-riding nodes employ a partial selfishness strategy. It would be interesting to see the performance of the scheme when dealing with nodes using such behaviour.

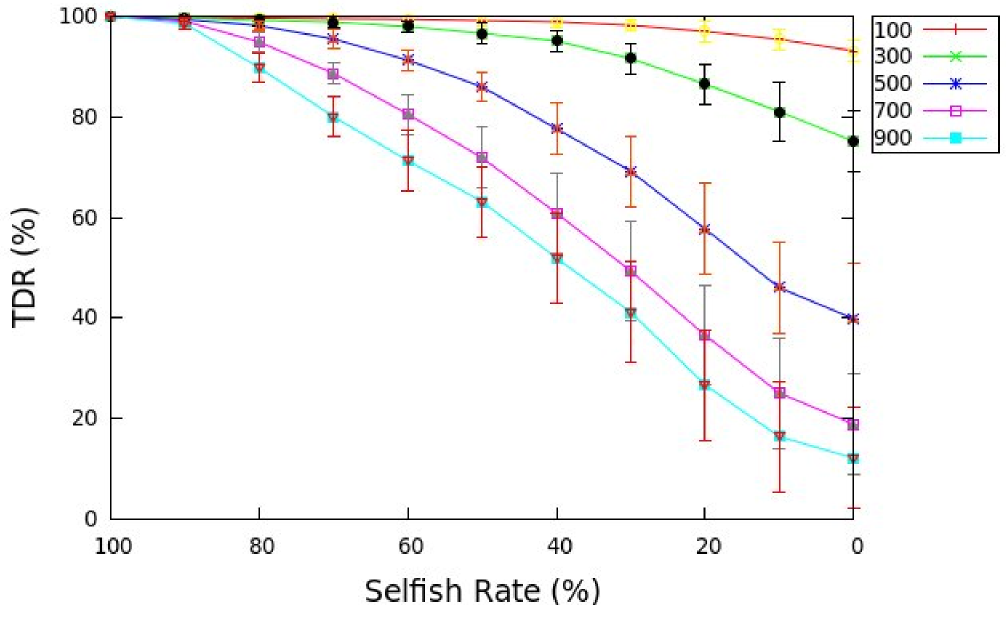
Free-riding nodes that employ partial selfishness only serve some percentage of the forwarding requests they receive and drop the other remaining forwarding requests. The ratio of the forwarding request that would be served by each free-riding node is indicated by a selfish rate percentage. For example, a free-riding node that employs 50% selfish rate means that it ignores half of the forwarding request it receives and forwards the other half.

In this experiment, we fix the mobility speed of each nodes at 2 m/s. We study the results of TDR and FDR as we vary the size of ActionHoldoffTime from 100 ms to 900 ms and selfish rate from 100% to 0%. The results presented for each data set are the average of 40 simulation runs, each with different seeds.

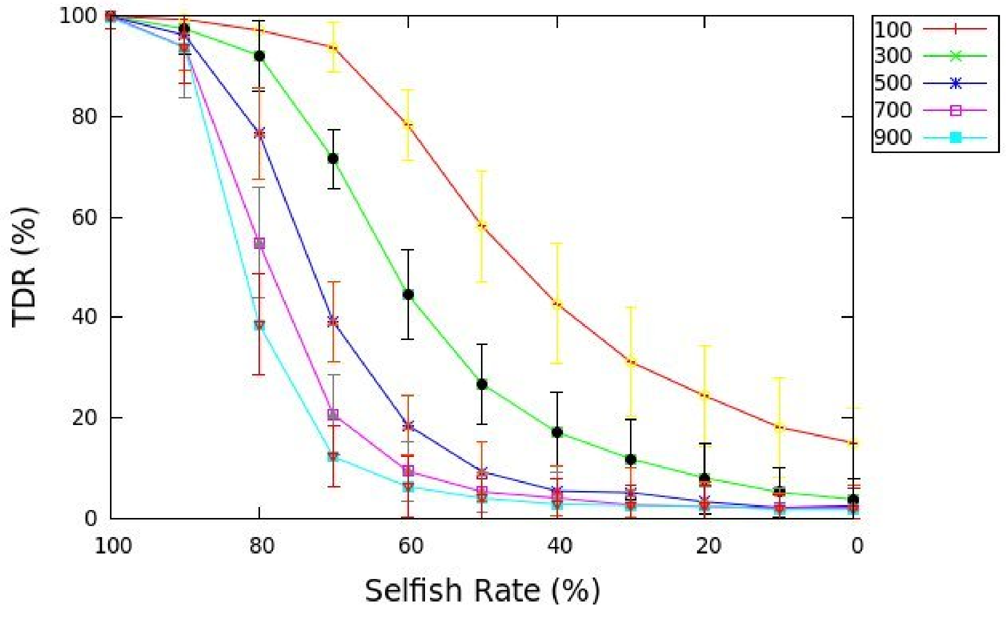
6.3.5.1 True Detection Ratio (TDR)

Figure 6.7a shows that the scheme is able to detect all free-riding nodes in the network when the nodes are employing 100% of selfish rate, regardless of the size of ActionHoldoffTime used. At 100% selfish rate, each free-riding node ignores all control and data packets that need to be forwarded to the next hop. When the monitoring nodes run suspicious checking, those free-riding nodes would fail the test and could be detected.

When the free-riding nodes behave in a lower selfish rate, the nodes serve some of the forwarding requests from their neighbouring nodes. The packets that are chosen to be forwarded could be good enough for the free-riding nodes to be seen consistently



(a) TDR without 10% Forgiveness



(b) TDR with 10% Forgiveness

Figure 6.7: TDR with Respect to Selfish Rate

contributing to the network. The packets could also be fake control packets from the suspicious checking procedure. In such scenarios, those free-riding nodes would be detected as well-behaving nodes and may defeat the detection scheme. As a result, the results of TDR will be decreased with a decreased value of selfish rate.

From Figure 6.7a, we also found that the results of TDR are largely influenced by the size of ActionHoldoffTime. For every set of selfish rate, the higher value of ActionHoldoffTime, the lower the result for TDR is. For example, at 50% selfish rate, when the size of ActionHoldoffTime is set to 100 ms, the result of TDR is 99%. But when the size of ActionHoldoffTime is set to 500 ms and 900 ms, the TDR decreases to 86% and 63% respectively.

In Figure 6.7b, we use a 10% forgiveness value. That means all nodes that are being monitored are allowed to drop 10% of the expected forwarding requests before the nodes are labelled as free-riding nodes. As depicted in the Figure, the use of this 10% forgiveness decreases the results of TDR compared to when 10% forgiveness is not used.

It is interesting to see the results of TDR when the free-riding nodes employ 0% selfish rate. At such a rate, the nodes practically do not behave as free-riding nodes. In other words, they are behaving as the well-behaving nodes and serve all forwarding requests. Ideally, the TDR for those scenario should be zero which implies that no free-riding nodes are detected. However, due to the packet collisions and the imperfect wireless channel, some of the forwarding requests are dropped and could not be served. As a results, these free-riding nodes that are employing 0% selfish rate are still detected as free-riding nodes. As we can see in Figure 6.7b, the use of 10% forgiveness improves the final results.

The results of TDR in Figure 6.7a when the free-riding nodes employ 0% selfish rate are slightly different from the results in Figure 6.2. This is because, in Figure

6.2, the results are based on the average of four scenarios: 0, 5, 10 and 15 free-riding nodes in a network which are operating at 100% selfish rate. In Figure 6.7a, the results are also based on the average of the four scenarios. However, in this case, all of the nodes are operating at 0% selfish rate. In other words, in the later Figure, all of the nodes are well-behaving.

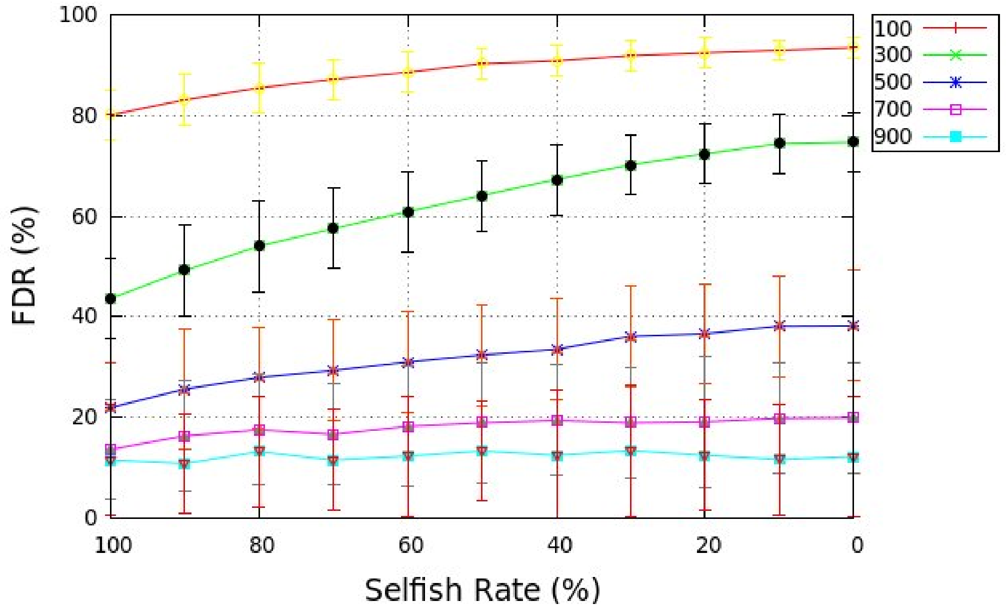
6.3.5.2 False Detection Ratio (FDR)

In this section, we evaluate the second performance metric, False Detection Ratio (FDR), when partial selfish rate is used. In Figure 6.8a, we found that in general, the higher we set the size of ActionHoldoffTime, the lower the results of the FDR. As we decrease the selfishness rate, the overall FDR increases. The rate of the increase is dependent on the size of ActionHoldoffTime. The higher the size of ActionHoldoffTime, the slower the increase of the FDR with respect to the selfish rate. For example, when the selfish rate is decreased from 100% to 50%, when the size of ActionHoldoffTime is 900 ms, the FDR increases by 5% compared to 12% when the size of ActionHoldoffTime is 500 ms.

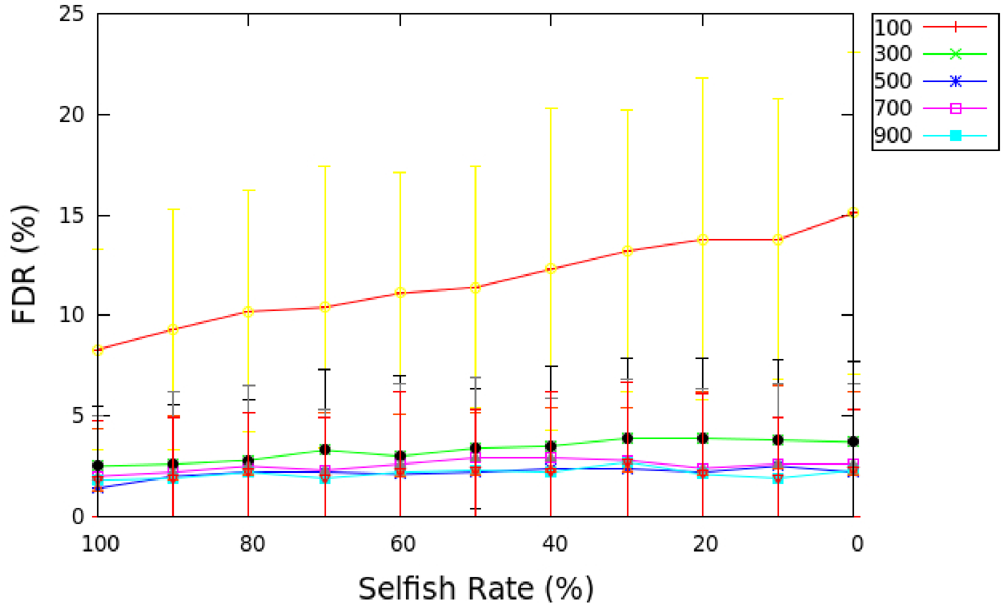
6.3.5.3 Effect of Suspicious Checking in Partial Selfishness

This section is related to section 6.3.4 where we evaluate the importance of suspicious checking in our detection scheme. This time, we discuss the issue with the respect to the partial selfishness strategy. In this experiment, we fix the mobility speed to 2 mps and the size of ActionHoldoffTime to 500. The selfish rate is in the range of 100% to 0%. The results presented for each data set are the average of 40 simulation runs, each with different seeds.

Figure 6.9a compares the results of FDR with and without the suspicious check-

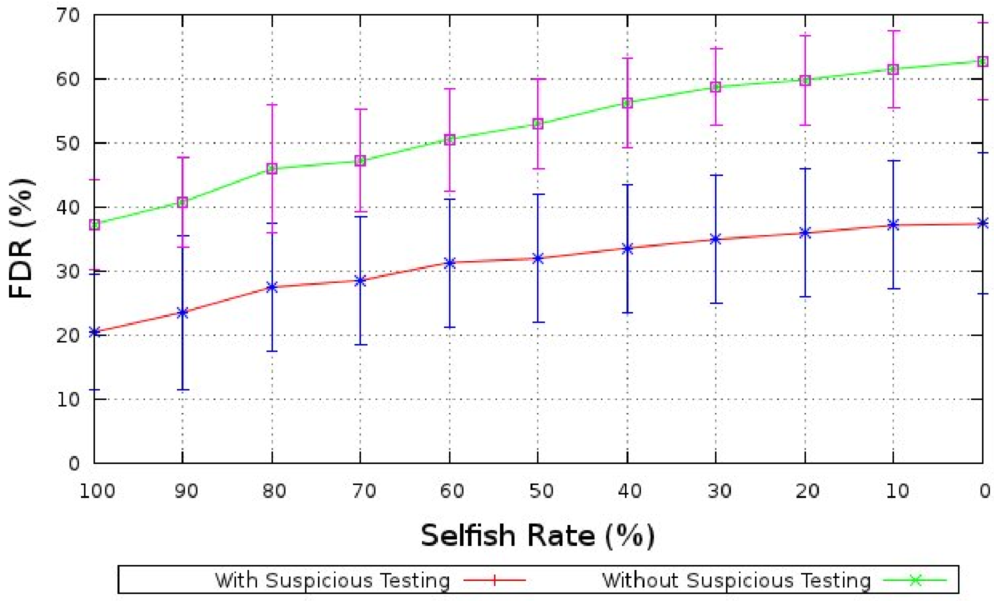


(a) FDR without 10% Forgiveness

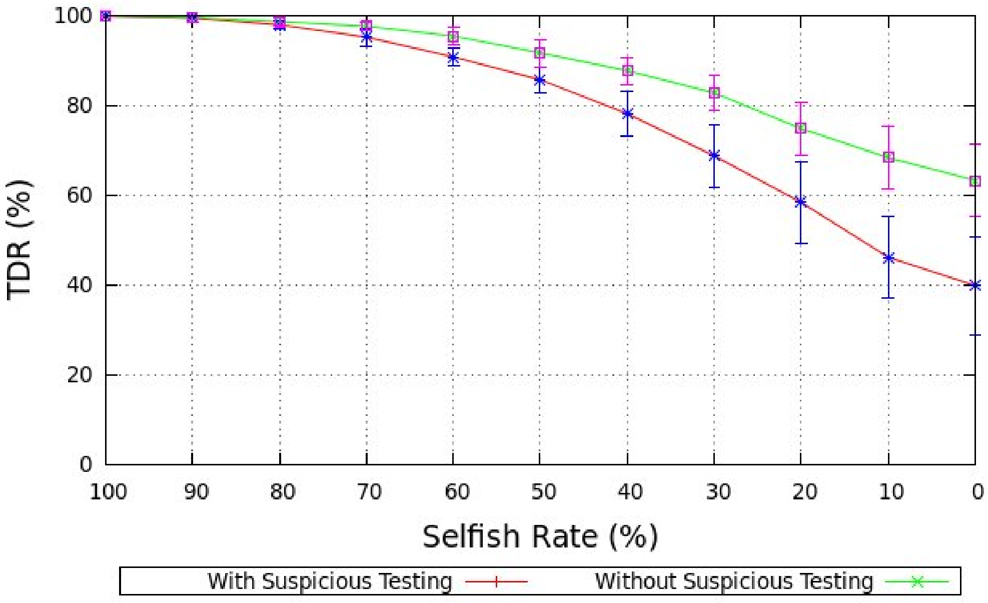


(b) FDR with 10% Forgiveness

Figure 6.8: FDR with Respect to Selfish Rate



(a) FDR



(b) TDR

Figure 6.9: Effect of Suspicious Checking with Partial Selfishness

ing procedure being used. As we can see, with suspicious checking, the FDR is always lower compared to when suspicious checking is not used. Suspicious checking improves the results of FDR in any rate of selfishness.

Figure 6.9b on the other hand, shows the results of TDR. When the free-riding nodes are behaving at the selfish rate of 100%, our detection scheme is able to detect all of the free-riding nodes in the network with or without the suspicious checking. As we decrease the selfish rate, the results of TDR is also decreased. However, when suspicious checking is used, the rate of this decrease is higher than that when suspicious checking is not used. So the results of TDR are always lower when suspicious checking is used.

One interesting result from the Figure is when the selfish rate set is 0%. At this point, the free-riding nodes are not behaving like one and not dropping any of the packets purposely. Theoretically the result of TDR should be 0%. Using the suspicious checking, the results are closer to 0% compared to when the suspicious checking is not used.

Suspicious checking gives the well-behaving nodes that are monitored a second chance to prove that they are not free-riding nodes. Due to the nature of the radio channel and also to the packet collision, some of the packets could not be captured by the monitoring nodes. This scenario could result the well-behaving nodes appearing as it they purposely drop the packets and could end up being accused as free-riding nodes. Using suspicious checking, the monitoring nodes would have to wait for the result of the checking before making any decision.

6.4 Summary and Conclusions

In this chapter, we present the results of our experiments that are described in the previous chapter. We use three performance metrics to evaluate our detection scheme namely True Detection Ratio (TDR), False Detection Ratio (FDR) and average routing overhead (RO). We evaluate the scheme by varying size of ActionHoldoffTime, the mobility speed and the number of free-riding nodes in the network. We also introduce 10% forgiveness ratio and study the effects of those ratio to the final detection results. Under the range of conditions given, the results of True Detection Ratio (TDR) are always 100%. The proposed scheme is able to detect all possible free-riding nodes in the neighbourhood. However, simulations show that the scheme does not achieve 0% of False Detection Ratio (FDR). Some of the well-behaving nodes are detected as free-riding nodes. The results of FDR depends on the size of ActionHoldoffTime, mobility speed and the number of free-riding nodes. A lower size of ActionHoldoffTime, higher mobility speed and lower number of free-riding nodes in the network contribute to higher results of FDR.

Since suspicious checking procedure increases the average routing overhead required, we also compare the performance of the detection scheme with when suspicious checking is not used. Performance evaluation of the scheme show that suspicious checking improves the results of FDR. However, the results of TDR are always 100% with or without the used of suspicious checking.

In the second part of the experiments, we evaluate our detection scheme when partial selfish rate is used. Again, we use the same performance metrics to investigate the effectiveness of the scheme. As the selfish rate employed by the free-riding nodes decreases, the results of TDR decreases and FDR increases. Fewer free-riding nodes are detected by the scheme. At the same time more well-behaving nodes are mistakenly detected as free-riding nodes.

There are several issues effecting the results of the detection scheme. Due to the poor quality of the network, some packets are lost during transmission. Packet collision and network congestion are the main factors of the packets lost. The packets may fail to be observed by the monitoring nodes which then make a wrong accusation to the benign nodes. The detection scheme may also has high FDR because of the mobility of the nodes. The topology of a MANET change dynamically due to the mobility of the nodes. Two nodes may not be within range all the time. A packet may not reach the intended node because by the time the packet is sent, the node move out of range. All of these issues contribute to the high false detection ratio of the scheme.

In the next chapter, we will discuss a proposed countermeasure that can be implemented after the free-riding nodes have been detected. The countermeasure or a method of punishment for the free-riding nodes will be used as an incentive for the nodes to be cooperative.

Chapter 7

Punishment Schemes for Free-riding Nodes

7.1 Introduction

The main objective of the punishment scheme is to minimize the effects of free-riding nodes in the network. As discussed in the previous chapter, free-riding nodes that act selfishly toward other nodes can decrease the performance of a MANET. Free-riding nodes utilize the packet forwarding services offered by cooperative nodes and send their packets be forwarded in a multi-hop communication. However, free-riding nodes themselves refuse to offer the services back to other nodes. As a result, the well-behaving nodes that cooperatively provide the services would have to bear the extra burden to forward the packets. This is not fair to the well-behaving nodes which have to use their scarce resources to provide the services to others including to free-riding nodes.

Free-riding nodes should not be allowed to continue with their behaviour without bearing any cost. One strategy to motivate the nodes to be cooperative is by showing that they would get better services (as an incentive) for being cooperative and get no service (as punishment) if they are not. However, it is difficult to implement this strategy in a open MANET where there is no centralised administration and no prior trust between the nodes. The punishment schemes have to be implemented carefully so that the well-behaving nodes executing the schemes themselves would not be detected as free-riding nodes and be punished by other nodes. In this chapter, we will discuss proposed punishment schemes that can be implemented in such condition. For simplicity, we use the term punishing node when referring to the well-behaving node that is running the punishment scheme. We also use the term monitoring nodes when referring to the well-behaving nodes that are monitoring their neighbouring nodes. For the context of this chapter, the punishing nodes are also the monitoring nodes since they operate both functions (monitoring and punishing) at the same time.

The remainder of the chapter is organised as follows: In section 7.2, the proposed punishment schemes are described in detail. In section 7.3, the performance metrics used to measure the performance of the schemes are explained. The description of the simulation parameters is presented in section 7.4 which is followed by the simulation results which are discussed in section 7.5. Finally, the chapter concludes with section 7.6.

7.2 Punishment Schemes Description

In this section, we will explain three variations of the proposed punishment scheme. Each of the monitoring node can be a punishing node. Since we are focusing on an open MANET, each of the participating node conducts the punishment scheme operation independently.

7.2.1 Scheme 1

The first proposed punishment scheme works as follows. For every packet that a node receives, it will first check if the packet is destined for itself. If that is the case, the node will grab the packet and deliver it to the upper layer (Transport Layer in OSI reference model) for further processing. If the packet is destined for another node, the punishing node will check further if the incoming packet is a data packet or control packet. The punishing node will forward the control packet to the next hop or rebroadcast the packet if it is a RREQ control packet.

Whenever a punishing node receives a data packet to be forwarded to the next hop, it will first capture the address of the source node where the packet originated from. The punishing node will check if the source node is one of the identified free-riding nodes in its table list. If the data packet originated from a free-riding node, the punishing node running the proposed punishment scheme will simply drop the packet. Otherwise, the data packet will be forwarded to the next hop. In short, a punishing node drops only data packets originated from a free-riding node but forwards the control packets. Figure 7.1 illustrates the steps in the punishment scheme 1.

In this scheme, the punishing nodes running scheme 1 still run the suspicious checking test on all of its neighbouring nodes when it is due.

7.2.2 Scheme 2

The second scheme is similar to the previous scheme. The punishing nodes drop data packets that originated from the free-riding nodes but still forward their control packets. However, unlike punishment scheme 2, the punishing nodes in this scheme only run the suspicious checking test for the nodes considered to be well-behaving

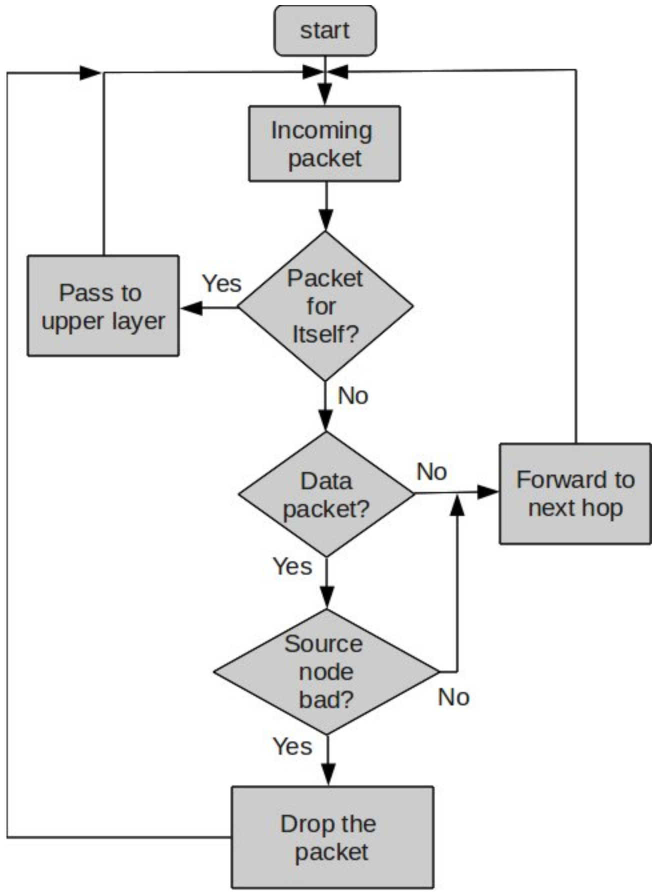


Figure 7.1: Flowchart for Punishment Scheme 1 and 2

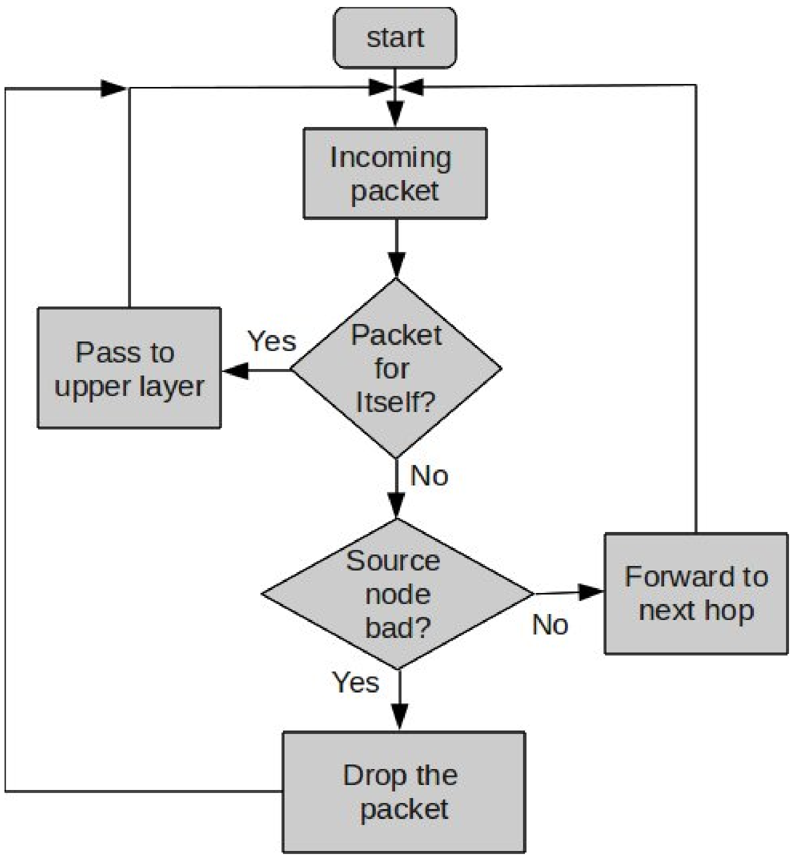


Figure 7.2: Flowchart for Punishment Scheme 3

nodes. Nodes that have been detected as free-riding nodes will not be tested.

7.2.3 Scheme 3

In the last scheme, the punishing nodes use a more extreme measure. Once the free-riding nodes have been identified, all packets originated from them including data and control packets will be dropped by the punishing nodes. Figure 7.2 illustrates the steps for punishing scheme 3. Similar to punishment scheme 2, the punishing nodes only run suspicious checking tests for the nodes that are not considered as free-riding nodes.

To make it easy for us to do the comparison, we refer the scheme where no pun-

Table 7.1: Summary of Punishment Scheme

	Packets drop from free-riding nodes	Suspicious test running
scheme 0	no dropping	to all nodes
scheme 1	only data	to all nodes
scheme 2	only data	only to well-behaving nodes
scheme 3	data and control	only to well-behaving nodes

ishment is implemented as scheme 0. Table 7.1 shows the summary of the described punishment schemes.

The only different between scheme 1 and scheme 2 is the scope of the suspicious checking. In scheme 1, suspicious checking is running for all nodes while in scheme 2, suspicious checking is running only for well-behaving nodes. As a result, in scheme 1, if the free-riding nodes decide to change their behaviour and become well-behaving nodes, they will have better chance not to be labelled as free-riding nodes anymore compared to in scheme 2.

Well-behaving nodes that are running any of the punishment schemes will not have greater chance of being accused as free-riding nodes by other well-behaving nodes. This is because the punishing nodes only punish the free-riding nodes and still provide forwarding services to other well-behaving nodes. The number of data packets forwarded successfully by the nodes running the any of the punishment schemes will be lower due to data packet dropping. However, that can be justified with the number of data packets that the punishing nodes forward for the well-behaving nodes.

7.3 Performance Analysis Metrics

When evaluating the proposed punishment scheme, we consider the following performance metrics.

7.3.1 Total Data Packet Forwarded (TDPF)

Total data packet forwarded is calculated using the following formula.

$$TDPF = \sum_{i=1}^{n} dp_i \tag{7.1}$$

dp : number of data packet forwarded by node i

n : the number of nodes in the network

We concentrate only on the total number of data packets forwarded by the well-behaving nodes that implement the punishment schemes.

7.3.2 Routing Overhead (RO)

Routing Overhead (RO) is calculated by the number of control packets sent over the number of data packets sent. Routing Overhead is calculated using the following formula:

$$RO = \sum_{i=1, d_i \neq 0}^{n} \frac{c_i}{d_i} \tag{7.2}$$

c : total number of control packets send from node i

d : total number of data packets send from node i

In this experiment, we want to focus on the routing overhead for well-behaving nodes when implementing the different punishment schemes.

7.3.3 Mean end-to-end delay

Mean end-to-end delay is used to access the quality of a network. Mean end-to-end delay refers to the average amount of time taken for the data packets to be transmitted across an MANET from source to destination. This includes all possible delay caused by buffering during route discovery latency, queuing at the interface queue, retransmission delay at the MAC, propagation and transfer times. The time is measured at the destination node each time a data packet arrived. For the purpose of this experiment, only the end-to-end delay for the data packets originated from the well-behaving nodes are measured. The formula to calculate mean end-to-end delay is as follows:

$$Mean\,Delay = \sum_{i=1,k_i\neq 0}^{n} \frac{t_r - t_s}{k_i} \tag{7.3}$$

$tr = received\ Time$

$ts = sent\ Time$

$k = number\ of\ data\ packets\ that\ arrived\ at\ node\ i$

$n = number\ of\ nodes\ in\ the\ network$

An ideal network should have the mean end-to-end delay as small as possible.

7.3.4 Packet Delivery Ratio

Packet delivery ratio (PDR) is another metric to measure the quality of a network. PDR refers to the ratio of the total number of data packets that reach the receiver (destination node) to the total number of data packets sent by the source node. It is calculated using the following formula:

$$PDR = \sum_{i=1, dt_i \neq 0}^{n} \frac{dr_i}{dt_i} \tag{7.4}$$

dr = number of data packets received at node i

dt = number of data packets transmitted from node i

n = number of nodes in the network

For a network, it is required that the packet delivery ratio is at a high level which indicates that most of the data packets arrive successfully at the destination.

7.4 Simulation Environment

The proposed punishment scheme for free-riding nodes in a MANET were simulated using the OMNET++ network simulator. For simplicity, we use the same simulation parameters that have been used in the previous experiments. The simulated network contained 20 mobile nodes which are randomly distributed in a playground area size of 500 m by 500 m. In our simulation environment, each of those nodes is both a source and destination for other nodes. All nodes in the network are transmitting and receiving data packets until the end of the simulation. For every data packet a source node sends, it could be for a different destination node. As a result, each source node may need to know the route to every other node in the network.

The MAC layer protocol IEEE 802.11 is used in all simulations with the data rate of 54 Mbps. The data packet size is 512 bytes. Traffic sources of constant bit rate (CBR) are used where the source node sends data packets with a frequency of 5 packets per seconds. The wireless transmission range of each node is 250 metres. The mobility speed is set to 2 m/s using a random way-point mobility model with

Table 7.2: Simulation parameters for Punishment Schemes

parameter	Value
Simulation Time	100 seconds
Number of Nodes	20
Packet Size	512 bytes
Network Area	500m x 500m
Mobility Model	Random WayPoint
Transmission Range	250 metres
Data Rate	5 packets/second
ActionHoldoffTime	500 ms
Mobility Speed	2 mps

a pause time of 0 second. We also fix the value for ActionHoldoffTime to 500 ms. Table 7.2 summarizes the parameters used in our simulation.

As with previous experiments, we evaluate our punishment scheme using the same four configurations of free-riding nodes in the networks which are listed below:

1. no free-riding node
2. 5 free-riding nodes
3. 10 free-riding nodes
4. 15 free-riding nodes

There is no significant importance to use network configuration where all 20 nodes are free-riding nodes. In such an environment, no node offers packet forwarding. There is no need to run a punishment scheme since no node can benefit from other nodes. In all simulation experiments in this chapter, we also do not concentrate on detecting free-riding nodes. We assume that each of the nodes already have the information about which nodes in the network are free-riding and need to be punished.

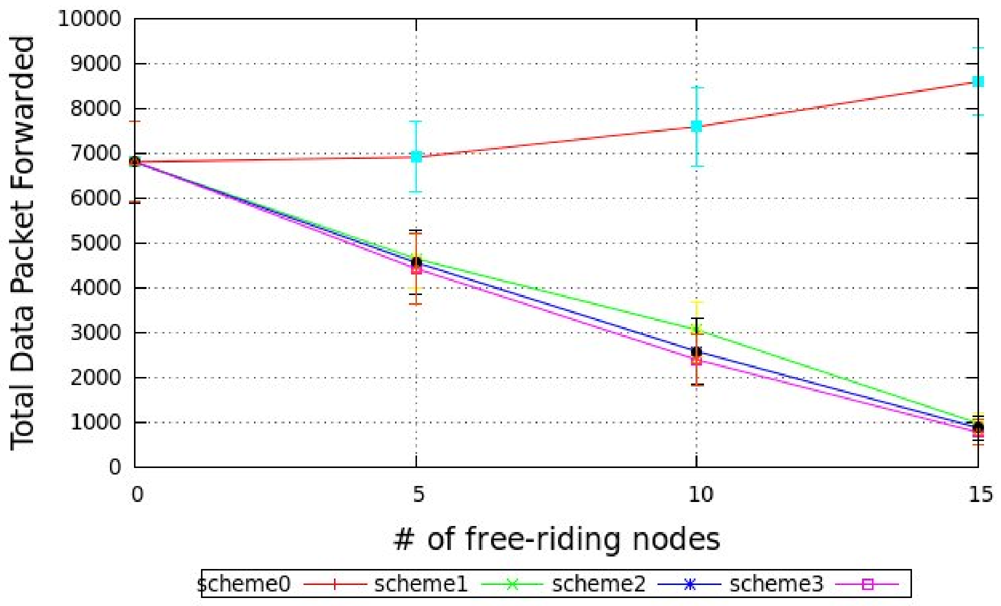


Figure 7.3: Total Data Packet Forwarded by Well-Behaving Nodes

7.5 Performance Evaluation

In this section, the simulation results from the experiments will be presented. The results presented for each data set are the average of 40 simulation runs, each with different seeds.

7.5.1 Total Data Packet Forwarded

Figure 7.3 shows the total number of data packets forwarded by well-behaving nodes with respect to the number of free-riding nodes in the network. Free-riding nodes will be relieved from forwarding data packets for other nodes while still sending their packets into the network. As a result, the well-behaving nodes may receive additional requests to forward data packets.

When there is no free-riding node in the network, no nodes will be punished. All nodes cooperatively forward data packets for each other. The total number of data packets forwarded by the nodes would be the same in any of the punishment schemes.

As the number of free-riding nodes in the network increases, there would be less option of routes for the data packets to reach the destination. The free-riding nodes excuse themselves from being part of the routing and would not be requested to forward data packets. When no punishment scheme is used, the well-behaving nodes will have to bear the extra burden to forward the data packets coming from the well-behaving nodes as well as from the free-riding nodes. As the number of free-riding nodes increases, the number of data packets forwarded by well-behaving nodes will be increased.

The punishment schemes allow the punishing nodes to refuse forwarding data packets for free-riding nodes. They only provide data packet forwarding service to other well-behaving nodes. So, as the number of free-riding nodes in the network increases, the number of the well-behaving would decrease. As a result, the number of data packets forwarded by the well-behaving nodes will decrease. This applies to punishment scheme 1, 2 and 3 where data packets from the free-riding nodes would be dropped and not forwarded by the punishing nodes.

7.5.2 Routing Overhead

As shown in Figure 7.4, when no punishment scheme is used, routing overhead increases as we increase the number of free-riding nodes in the network. As the number of free-riding nodes increases, the number of suspicious checks that are conducted increases which contributes to the increased routing overhead of the well-behaving nodes.

In punishment scheme 1, the punishing nodes drop data packets originating from

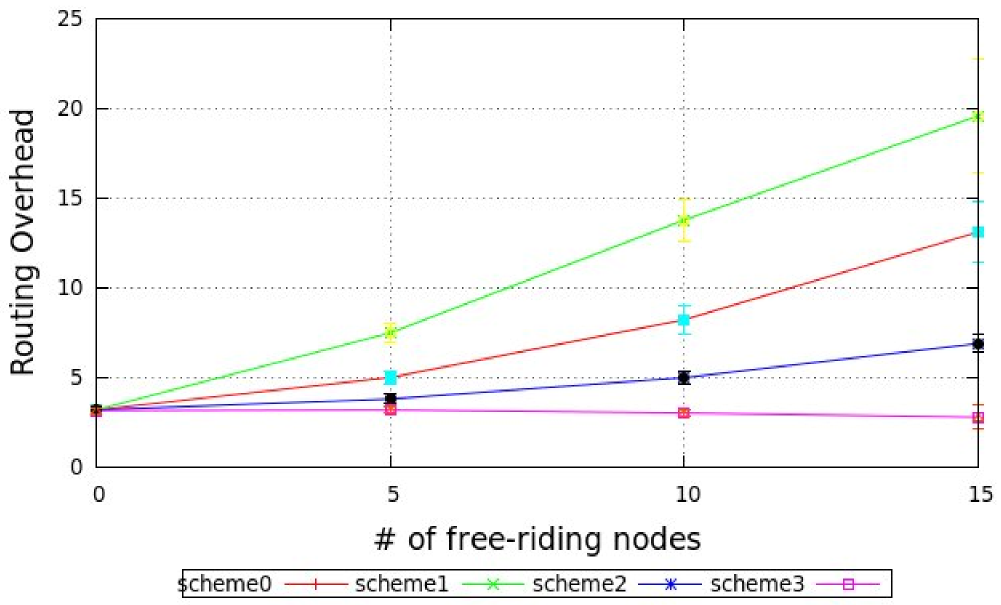


Figure 7.4: Routing Overhead by Well-Behaving Nodes

free-riding nodes but still forward their control packets. When the free-riding nodes do not get the acknowledgement for the packets they send to be forwarded, they will retransmit the data packet. This causes the punishing nodes to run more suspicious checks for the free-riding nodes. During suspicious checking, the monitoring nodes inject additional RREQ control packets which also cause other neighbouring nodes to rebroadcast the packet into the network. Again, more suspicious checking means more packets are injected which contributes to the increase routing overhead of the well-behaving nodes.

In punishment scheme 2, the punishing nodes do not run suspicious checking and that reduces the routing overhead used compared to scheme 0 and 1. In punishment scheme 3, the punishing nodes do not respond to control packets from free-riding nodes. The punishing nodes will neither have to forward data packets for free-riding nodes nor have to run suspicious checking testing for them. The increased number of free-riding nodes in the network would not affect the total routing overhead for the punishing nodes.

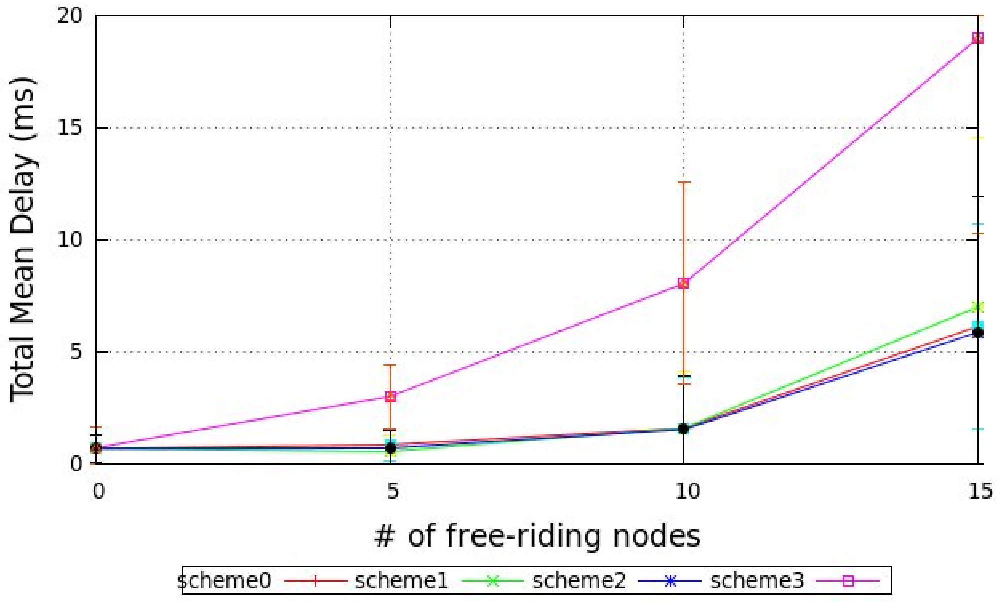


Figure 7.5: Total Mean End-to-End Delay

7.5.3 Total Mean End-to-End Delay

Figure 7.5 depicts the effects of the proposed punishment schemes on the total mean end-to-end delay for a network. As we increase the number of free-riding nodes in the network, the total mean end-to-end delay also increases in each of the punishment schemes. The total mean end-to-end delay for punishment scheme 1 increases almost at the same rate as punishment scheme 2 but slightly higher compared to when no punishment scheme is used. However, punishment scheme 3 causes the total mean end-to-end delay to increase much further compared to the other two schemes.

In schemes 1 and 2, the free-riding nodes are able to find the route to the destination for their data packets. However, if the next hop in the route contains the punishing node, the data packets will be dropped and will not reach the destination. The free-riding nodes then have to retransmit the data and execute another route discovery after several trials to find other routes to the destination. In this case, the free-riding nodes have to wait until the the destination node moves within their

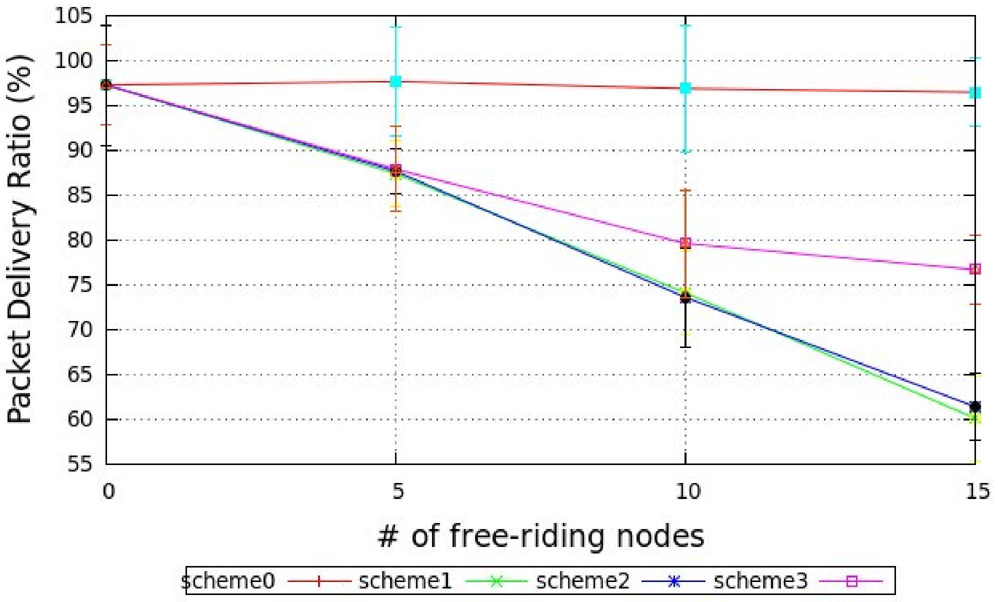


Figure 7.6: Packet Delivery Ratio in Punishment Schemes

transmission range.

The nodes running punishment scheme 3 ignore all packets originating from the free-riding nodes. As a result, a free-riding node will not be able to establish multi-hop communication through any of the punishing nodes. Since other free-riding nodes will also not be a part of the routing, the only way for the free-riding nodes to send their packets is through one-to-one direct communication. The free-riding nodes will have to run route discovery until the source and the destination nodes are within the transmission range. The lack of routes to the destination increases the queue waiting time in the existing available routes which then causes the total mean end-to-end delay for the network to increase.

7.5.4 Packet Delivery Ratio (PDR)

Figure 7.6 illustrated the packet delivery ratio when different punishment schemes are used with respect to the number of free-riding nodes in a network. Packet delivery ratio is not affected by the number of free-riding nodes when no punishment scheme is used. This result tallies with the analysis on the effect of free-riding nodes which has been discussed in the chapter earlier. Data packet forwarding is provided by the well-behaving nodes in the network.

In schemes 1 and 2, well-behaving nodes drop data packets originating from free-riding nodes. As the number of free-riding nodes in the network increases, more data packets are dropped by the punishing nodes which reduces the packet delivery ratio. In scheme 3, the punishing nodes ignore control packets originating from the free-riding nodes. The punishing nodes will not be selected to forward data packets for the free-riding nodes. Data packets from the free-riding nodes can only reach the destination nodes if they are within range. As a result, the packet delivery ratio is not as low compared to when scheme 1 or 2 is used.

7.6 Conclusion

Free-riding nodes that have been detected need to be motivated so that they would not continue with their behaviour. One way to establish the motivation is to give better services to the well-behaving nodes and to impose some sort of punishment to the free-riding nodes. The proposed punishment schemes described in this chapter involve denying data forwarding services for the free-riding nodes.

When no punishment scheme in place, the data forwarding will be performed by only the well-behaving nodes. Free-riding nodes will be relieved from forwarding data packets but will still get their their own packets be forwarded by others. The more number of free-riding in the network, the more well-behaving nodes have to take the burden of forwarding the data packets.

In this chapter, we propose three variations of punishment schemes that can be used in a open MANET where each of the participating nodes works independently. In punishment scheme 1, the punishing nodes punish free-riding nodes by dropping their data packets. Nevertheless, control packets from free-riding nodes are still forwarded by the punishing nodes. In punishment scheme 2, the punishing nodes also drop data packets originated from the free-riding nodes. However, unlike previous scheme, the punishing nodes stop running the suspicious checking testing for the free-riding nodes. In punishment scheme 3, the punishing nodes isolate the free-riding nodes by ignoring all of the packets including control packets that originated from them.

All three variations of the punishment schemes are simulated using OMNET++. Their performance are compared with each other and with when no punishment scheme is used, in term of total data packet forwarded, routing overhead, mean end-to-end delay and packet delivery ratio.

Chapter 8

Summary, Conclusions and Future Work

8.1 Summary

A Mobile Ad hoc Network (MANET) is a wireless network that is formed automatically by a collection of mobile nodes without relying on a fixed infrastructure or centralised management. MANETs have been an area for active research over the past few years. This has been motivated by recent advances in wireless technology and mobile computing devices and the enormous applications that could be realised using MANETs. MANETs offer unique benefits and greater flexibility compared to their counterparts in traditional wired and wireless networks. Nevertheless, they present particular challenges arising mainly from the absence of centralised administration, dynamic network topology and capacity constrained operation.

One of the unique characteristics of a MANET is the ability to establish multi-hop communication. Two nodes in a MANET are able to communicate beyond the transmission range by relying on the intermediate nodes to cooperatively relay the messages hop by hop. This cooperation can be guaranteed if all of the nodes belongs to a single authority and pursue a same goal. However, in an open MANET, the nodes belong to different authorities or individuals. In such an environment, the nodes may have different goals and have little incentive to use up energy to forward packets for other nodes. If a large number of nodes choose to behave in this selfish manner, a MANET would degrade into a network in which the nodes can only communicate with other nodes within their direct communication range, thereby defeating the very purpose of forming a MANET.

The research presented in this thesis addresses the effects of selfish or misbehaving nodes on the performance of a MANET. The aim of the misbehaving nodes may not be to launch an attack but to obtain an unfair advantage from other nodes. They use forwarding services provided by other nodes but to save their own resources, refuse to offer the service back. There are two types of misbehaving nodes: misleading and free-riding nodes. Misleading nodes forward control packets from other nodes but ignore the requests when they receive data packets to be forwarded. Free-riding nodes drop all packets (control and data) that are not destined for them. As a result, they do not receive requests to forward data packets. In chapter 4, the discussion about the two types of misbehaving nodes and their effects on the performance of a MANET are presented.

The major focus of this research has been the proposal and analysis of new schemes to deal with free-riding nodes in an open MANET. In chapter 5, we propose a detection scheme which adopts a widely used mechanism known as the watchdog scheme. In the proposed detection scheme, each node in the network independently monitors activities of its neighbouring nodes by analysing the packets it receives through promiscuous mode. The participating nodes are expected to contribute to the network continuously within a predefined period of time. If some of them fail to do so, a suspicious checking procedure would be conducted to give them another chance to contribute to the network before they are labelled as free-riding nodes.

The evaluation of the detection scheme and the simulation results are presented in chapter 6.

The free-riding nodes that have been detected should bear the cost for their selfishness. In chapter 7, we propose three variations of punishment schemes that can be executed on the free-riding nodes. In the first and the second punishment schemes, the punishing nodes (nodes that running punishment scheme) drop data packets originating from the free-riding nodes. However, in second scheme, no suspicious checking is conducted to the free-riding nodes. In the third punishment scheme, the punishing nodes drop all packets originating from the free-riding nodes. Similar to the second scheme, no suspicious checking is conducted for the free-riding nodes. All of the punishment schemes are evaluated and compared to see their effects on the free-riding nodes as well as on the overall network.

8.2 Conclusions

In order to implement the proposed schemes, a simulation model was developed. The simulation model developed using OMNET++ was essential in proving the working of the proposed schemes. In this section, conclusions will be drawn based on the results obtained from the simulation model carried out in previous chapters.

8.2.1 Design and Implementation of the Free-riding Detection Scheme

Most of the schemes to detect misleading nodes only monitor the number of data packets dropped. They are therefore not suitable to be used to detect free-riding nodes. Free-riding nodes excuse themselves from being part of the routing and as a result are not expected to forward data packets. The proposed detection scheme also provides fairness to all participating nodes in the network. Only nodes that request service from the network need to contribute back. Nodes that only use single hop communication to send packets, or are inactive, do not use the services from other nodes and thus are not obliged to contribute.

The proposed detection scheme allows each of the monitoring nodes to operate the detection independently. In the proposed detection scheme only first-hand observation is used to make decisions. First-hand observation includes personal experience and direct observation. When making a decision on the detection, the monitoring nodes do not use other nodes' opinions. None of the decisions are shared with other nodes. One advantage of this approach is that we do not have to handle the complexity of disseminating the opinion from one node to another. In addition, we can avoid collusion where two or more nodes collaborate with each other to accuse innocent nodes or falsely praise each other in order to defeat the detection scheme.

We take into consideration that some of the nodes in the network may not have a chance to forward data for others. One example is the nodes that are located at the edge of the network. At that location there are no other nodes to send data packets to. Even in a scenario where there are many options of routes, there is also no guarantee that a particular node will always be selected to forward data packets. Those nodes are well-behaving nodes and should not be labelled as free-riding nodes.

Another issue that needs to be considered is the quality of the network. Due to the wireless channel condition degradation (e.g., packet collision and congestion), some packets are lost during transmission. The node that is being monitored may actually forward a packet but fail to be observed by the monitoring node. The packet from the source node may also fail to reach the intermediate node which then can not respond to the forwarding request. The same scenarios can also be caused by the mobility of the nodes. Due to the mobility, there is no guarantee that the nodes that are being monitored and the monitoring nodes are always within range. Some of the packets may not reach the intended nodes because by the time the packets are sent, the nodes move out of range. All of these scenarios cause the monitoring nodes to make wrong observations and affect the false detection ratio of the detection scheme.

For these unfortunate nodes, we introduce a special test called suspicious checking where fake RREQ control packets will be injected to the network to give those nodes a chance not to be labelled as free-riding nodes. From the simulations, we can see that suspicious checking improves the detection scheme by reducing the number of well-behaving nodes from being accused as free-riding nodes. Depending on the mobility speed of the nodes, the false detection ratio can be improved up to 38% by using the suspicious checking.

The drawback of the suspicious checking is that the routing overhead of the network will be higher. Monitoring nodes running direct observation inject fake RREQ control packets when testing the suspicious nodes. The fake RREQ control packets are also forwarded by other nodes in the neighbourhood increasing the routing overhead. From the simulations, we can see that overall routing overhead by the monitoring nodes can be increased by up to 85% compared to when the test is not implemented. In other words, the proportion of control traffic to data traffic in the network increases when suspicious checking is used.

In the last part of Chapter 6, we have also evaluated our detection scheme to handle a strategy called partial selfishness. In partial selfishness, free-riding nodes forward some percentage of the packets and drop the remaining. From the simulations, we found that the results of the proposed detection scheme are greatly influenced by the time windowing size called ActionHoldoffTime. ActionHoldoffTime is the period of time a node needs to contribute back to the network before further action is taken by a monitoring node. The results for true detection ratio are better when the size of ActionHoldoffTime was smaller. However, the smaller size of ActionHoldoffTime gives worse results for false detection ratio.

There are some limitations of the proposed detection scheme. Even though our scheme is not dependant on third party information when detecting free-riding nodes, it still relies on the suspicious checking that is expected to be conducted by other nodes. As mentioned before, there are two kinds of monitoring used in our scheme: personal experience and direct observation. In the proposed detection scheme, the monitoring nodes expect all of their active neighbouring nodes to continuously contribute to the network in every predefined time period. If for any reason one of the neighbouring nodes fails to do so, suspicious checking is assumed to be conducted by the node that receives the data forwarding request. However, if the relevant node refuses to conduct the suspicious testing, the suspicious node will never have a second chance to contribute to the network.

There are two possible reasons for the relevant monitoring node not to run suspicious checking. One reason is because to the relevant node, the time for the suspicious node to be tested has not yet been reached. Since each of the monitoring nodes is working independently, some of them may have different values of last_request and last_action for the same node. In MANETs, the nodes are free to move in an arbitrary fashion. One node may be within range of a monitoring node in one time and then move out of range a moment later. During the simulation, the period of time a single node moves within range with different monitoring nodes would not be the same. As a result, the time to conduct suspicious checking for a node may be different for different monitoring nodes.

Another reason for the relevant node not to run suspicious checking is simply for saving energy. As explained before, sending packets consumes energy. The relevant node may not wish to participate in the detection scheme and see no benefit in conducting suspicious checking. If either of the two scenarios happens, the unfortunate node will be labelled as free-riding. Since there is no information sharing between the monitoring nodes and the fake RREQ control packet is supposed to be anonymous, there is no way the monitoring nodes running direct observation could determine if suspicious checking has been conducted beforehand.

Another important concern in running the detection scheme is the power consumption. The monitoring nodes have to operate in promiscuous mode and additional packets have to be injected into the network during suspicious checking. Running promiscuous mode is expensive since more packets are going to the network layer and more energy is needed to process the packets. In addition, injecting packets into the network also consume energy as been explained before. One would argue whether it is worth to run detection scheme considering all the above reasons.

The author believes that it is vital to have prevention strategy to maintain the productivity and the healthtiness of an open MANET. If the free-riding nodes are not detected and punished, there would be no reward for any nodes to cooperate. A node that may be initially choses to cooperate will find itself have to serve other nodes that are not cooperative. Without any motivation, the node may see no benefit to contribute and decide to stop cooperate. At the end, all nodes will choose to become free-riding. As the quality of the open network deteriorates, all nodes may leave the network gradually. In this thesis, the author propose a scheme that could avoid the situation to happen by detecting and penalizing the free-riding nodes.

8.2.2 Design and Implementation of the Free-riding Punishment Scheme

Punishment schemes forbid the free-riding nodes from getting advantages for being selfish. In this thesis, we introduce three variations of punishment schemes. Those three schemes are modelled in the network simulations and compared with each other. They are also compared with when no punishment scheme is implemented.

In any of the punishment schemes, only data packets forwarding requests from other well-behaving nodes will get the service. The free-riding nodes will not have their data packets forwarded by the well-behaving nodes which are now operating as punishing nodes. The only way for the free-riding nodes to have their data sent to other nodes is by direct communication where the source and the destination nodes have to be within range. As the percentage of free-riding nodes increases in the network, the well-behaving nodes have fewer genuine requests to forward data packets.

The average routing overhead for the well-behaving nodes resulting from running the three variations of punishment schemes is different. The routing overhead is higher in scheme 1, followed by scheme 2 and scheme 3. Much of the overhead comes from the suspicious checking which is still conducted for every node in scheme 1. In scheme 2, the overhead is smaller because the punishing nodes do not run suspicious checking for the free-riding nodes. The overhead is lowest in scheme 3. The punishing nodes do not run suspicious checking and also ignore packets from free-riding nodes.

However, punishment schemes decrease the overall network performance. Total mean end-to-end delay increases dramatically especially in scheme 3 as the number of free-riding nodes increases. Since the punishing nodes drop data packets originating from free-riding nodes, packet delivery ratio (PDR) of the network decreases. Fewer data packets are able to reach the destination nodes compared to when no punishment scheme is running. A free-riding node can only send its data when the destination is within a single hop communication range.

8.3 Future Work

A few issues can be considered for further research. In this work, the detection and punishment schemes operate separately. Prior to the executing of the punishment scheme, it is assumed that the free-riding nodes have already been detected. For the future work, we believe detection and punishment scheme should be integrated when dealing with the free-riding nodes. In doing so, the detection scheme should be improved as to have zero false positive detection ratio. There should also be a decision on when to make the final judgement in the detection scheme before the punishment scheme is implemented. It could either be for each period of time the monitoring nodes have been operating or just based on the number of times the monitoring node has been interacting with the monitored node. Another issue that should be of concern is the way to handle a node that is well-behaving in the beginning and then starts to behave as free-riding later on. All of these issues should be handled when designing the integration of detection and punishment schemes.

One of the assumptions that we use in this project is that each node will have an identity which is persistent, unique and distinct. This is important to ensure that the nodes that are monitored and punished are the actual nodes. However, in mobile ad hoc network, the identity of the nodes can be easily obtained, changed or discarded given that there is no centralised identity management. As there is no restriction on identity creation, a node can create as many identities as it wishes. One example of attacks in this category is whitewashing. In this identity changing attack, a node that has a previous misbehaving history leaves the network and rejoins using a different identity to avoid being detected and punished. There are two ways that have been proposed in the literature to counter whitewashing attack [71]. The first is to require the use irreplaceable pseudonyms, e.g., through assignment of strong identities by a central trusted authority that does not exist in open MANETs. The second one is to impose a penalty on all newcomers, including both legitimate newcomers and whitewasher (a node that is running whitewashing attack). The second technique is more suitable to be used in a MANET because it does not need any centralised trusted third party for identity management.

During the proposed suspicious checking, the monitoring node sends a fake RREQ control packet to give a chance to a suspicious node to contribute to the network. In this work, the control packet is designed in a simple manner. We use a nonexist address for the destination field by incrementing the third octet of the targeted IP address. To minimize the increase of routing overhead in the network, we restrict the number of hops the packet should travel by setting the value of its TTL to 1. With these unique characteristics, it is understandable that node that receives the control packet can easily distinguish the fake RREQ packet from the normal packets. Free-riding nodes may only respond to such packets which then defeat the proposed detection scheme. For the future work, we believe that an alternative way to design fake RREQ control packets should be studied. The new design of the packet should be similar to normal packets and at the same time handle the routing overhead of the network.

In our detection scheme, we use the same weight for amount of data forwarded and number of suspicious tests passed to determine whether or not a monitored node is a free-riding node. Based on the simulation, the first element of the formula is too big compared to the second element. It would be interesting to see if putting more weight on suspicious tests passed compared to number of data packets forwarded can reduce the false detection ratio (FDR) of the scheme.

We have shown the working and implementation details of the detection and punishment schemes for open MANETs running on top of the DSR routing protocol. For future work, we would like to extend the application of the scheme to other routing protocols and domains.

Bibliography

[1] J. Hoebeke, I. Moerman, B. Dhoedt, and P. Demeester, “An overview of mobile ad hoc networks: applications and challenges,” in *Journal of the Communications Networks*, vol. 3, pp. 60–66, 2004.

[2] I. Chlamtac, M. Conti, and J. J. Liu, “Mobile ad hoc networking: imperatives and challenges,” in *Ad Hoc Networks*, vol. 1, p. 1364, 2003.

[3] C. Perkins, *Ad-hoc Networking.* Addison-Wesley, 2001.

[4] V. K. Garg, *Principles and applications of GSM.* Upper Saddle River (NJ) Prentice Hall PTR, 1999.

[5] S. Kasera and N. Narang, *3G Mobile Networks: Architecture, Protocols, and Procedures.* McGraw Hill Professional Engineering, 2004.

[6] J. Dunlop, D. Girma, and J. Irvine, *Digital Mobile Communications and the TETRA System.* John Wiley & Sons, 1999.

[7] “IEEE standard 802.11-1999,” September 1999.

[8] Bluetooth, “Specification of the bluetooth system.” Available at https://www.bluetooth.com, February 2001.

[9] M. S. Ltd, “Introduction to ZigBee.” http://www.measurementsystems.co.uk/p59/Introducti to-ZigBee/pages.html. [accessed on Feb 7, 2010].

[10] S. Farahani, *ZigBee Wireless Networks and Transceivers.* Elsevier, 2008.

[11] H. Miranda and L. Rodrigues, “Preventing selfishness in open mobile ad hoc networks,” in *Proc. of the Seventh CaberNet Radicals Workshop*, October 2002.

[12] E. H. Callaway, *Wireless Sensor Networks: Architectures and Protocols.* Auerbach Publications, 2003.

[13] J. Garcia-Macias and J. Gomez, *Sensor Networks and Configuration*, ch. MANET versus WSN, pp. 369–387. Springer, 2007.

[14] I. Chlamtac and A. Lerner, "Fair algorithms for maximal link activation in multihop radio networks," *IEEE Transactions on Communications*, vol. 35, pp. 739–746, July 1987.

[15] I. Chlamtac and A. Lerner, "Link allocation in mobile radio network with noisy channel," in *IEEE INFOCOM*, April 1986.

[16] J. A. Freebersyser and B. Leiner, "A DoD perspective on mobile ad hoc networks," in *Ad hoc networking* (C. E. Perkins, ed.), pp. 29–51, Addison-Wesley, 2001.

[17] N. Vetrivelan and A. V. Reddy, "Analysing the mobility and protocol performance in manet using a novel move stop deviate model," *Mobile Communication*, vol. 4, pp. 75–80, 2010.

[18] E. M. Royer and C.-K. Toh, "A review of current routing protocols for ad-hoc mobile wireless network," *IEEE Personal Communications*, vol. 6, pp. 46–55, Apr 1999.

[19] P. Chandra, *Bulletproof Wireless Security: GSM, UMTS, 802.11 and Ad Hoc Security.* Elsevier, 2003.

[20] M. Gunes and O. Spaniol, "Routing algorithms for mobile multi-hop ad hoc," 2002.

[21] D. B. Johnson, D. A. Maltz, and J. Broch, "DSR: the dynamic source routing protocol for multihop wireless ad hoc networks," in *Ad Hoc Networking* (C. E. Perkins, ed.), pp. 139–172, Addison-Wesley, 2001.

[22] D. B. Johnson, "Routing in ad-hoc networks of mobile hosts," in *Workshop on Mobile Computing Systems and Applications*, (Santa Cruz, CA, US), Dec 1994.

[23] J. Costa-Requena, *A Hybrid Routing Approach for Ad Hoc Network*. PhD thesis, Helsinki University of Technology, Networking Laboratory, 2007.

[24] J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *Mobile Computing and Networking*, pp. 85–97, 1998.

[25] T. Sundararajan and A. Shanmugam, "Performance analysis of selfish node aware routing protocol for mobile ad hoc networks," *ICGST International Journal on Computer Network and Internet Research (CNIR)*, vol. 9, no. 1, pp. 1–9, 2009.

[26] Y.-C. Hu, D. B. Johnson, and D. A. Maltz, "The dynamic source routing protocol for mobile ad hoc networks (dsr)." http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-09.txt, April 2003.

[27] C. D. M. Cordeiro and D. P. Agrawal, *Ad Hoc & Sensor Networks: Theory and Applications*. World Scientific Publishing Company, 2006.

[28] 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)*, January 2002.

[29] S. Yi and R. Kravets, "Composite key management for ad hoc networks," in *First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'04)*, pp. 52–61, 2004.

[30] H. Yang, H. Luo, F. Ye, S. Lu, and L. Zhang, "Security in mobile ad hoc networks: Challenges and solutions," *IEEE Wireless Communications*, vol. 11, pp. 38–47, 2004.

[31] K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer, "A secure routing protocol for ad hoc networks," in *10th IEEE International Conference on Network Protocols*, pp. 78–87, November 2002.

[32] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Packet leashes: A defense against wormhole attacks in wireless networks," in *Twenty-Second Annual Joint Conference of the IEEE Computer and Communications*, vol. 3, pp. 1976–1986, IEEE Societies, April 2003.

[33] Y.-C. Hu, A. Perrig, and D. B. Johnson, “Ariadne: A secure on-demand routing protocol for ad hoc networks,” in *MobiCom ’02*, September 2002.

[34] L. Buttyan and J.-P. Hubaux, “Security and cooperation in wireless networks.” http://secowinet.epfl.ch, July 2007.

[35] R. Zheng and R. Kravets, “On-demand power management for ad hoc networks,” in *Ad Hoc Networks*, vol. 3, pp. 51–68, January 2005.

[36] L. Rasmusson and S. Jansson, “Simulated social control for secure internet commerce,” in *Proc. of the 1996 New Security Paradigms Workshops, ACM*, pp. 18–26, 1996.

[37] L. Eschenauer, V. D. Gligor, and J. Baras, “On trust establishment in mobile ad-hoc networks,” in *Proc. of the Security Protocols Workshop*, pp. 47–66, Springer-Verlag, 2002.

[38] A. Abdul-Rahman and S. Hailes, “A distributed trust model,” in *Workshop on New Security Paradigms, Langdate, UK*, pp. 48–60, 1997.

[39] A. Abdul-Rahman and S. Hailes, “Supporting trust in virtual communities,” in *33rd Annual Hawaii International Conference on System Sciences (HICSS 33)*, vol. 1, pp. 6007–6016, January 2000.

[40] D. Gambetta, “Can we trust trust?,” in *Trust: Making and Breaking Cooperative Relations*, pp. 213–237, Basil Blackwell, 1988.

[41] L. Buttyan and J.-P. Hubaux, “Enforcing service availability in mobile ad-hoc wans,” in *IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC)*, (Boston), August 2000.

[42] L. Buttyan and J.-P. Hubaux, “Stimulating cooperation in self-organizing mobile ad hoc networks,” in *Mobile Networks and Applications*, vol. 8, pp. 579–592, October 2003.

[43] S. Zhong, J. Chen, and Y. R. Yang, “Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks,” in *INFOCOM 2003*, 2003.

[44] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehaviour in mobile ad hoc networks," in *Proc. of Mobile Computing and Networking (MobiCom'00)*, pp. 255–265, August 2000.

[45] P. Michiardi and R. Molva, "CORE: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks," in *Proc. of Sixth IFIP Conference on Communication and Multimedia Security (CMS'02)*, September 2002.

[46] S. Bansal and M. Baker, "Observation-based cooperation enforcement in ad hoc networks," tech. rep., Stanford University, 2003.

[47] S. Buchegger and J.-Y. L. Boudec, "Performance analysis of the CONFIDANT protocol: (cooperative of nodes - fairness in dynamic ad hoc networks)," in *Proc. of IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHoc'02)*, pp. 226–336, June 2002.

[48] K. Balakrishnan, J. Deng, and V. Varshney, "Twoack: Preventing selfishness in mobile ad hoc networks," in *Wireless Communications and Networking Conference*, vol. 4, pp. 2137–2142, 2005.

[49] K. Liu, J. Deng, P. K. Varshney, and K. Balakrishnan, "An acknowledgment-based approach for the detection of routing misbehavior in manets," in *IEEE Transactions on Mobile Computing*, pp. 536–550, 2006.

[50] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens, "An on-demand secure routing protocol resilient to byzantine failures," in *ACM Workshop on Wireless Security (WiSe)*, September 2002.

[51] F. Kargl, A. Klenk, S. Schlott, and M. Weber, "Advanced detection of selfish or malicious nodes in ad hoc networks," in *1st European Workshop on Security in Ad-Hoc and Sensor Networks (ESAS 2004)*, Springer, 2004.

[52] G. Jager, "Game dynamics connects semantics and pragmatics," in *Game Theory and Linguistic Meaning* (A.-V. Pietarinen, ed.), pp. 89–102, Elsevier, April 2008.

[53] J. F. Nash, "Equilibrium points in n-person games," in *Proceedings of the National Academy of Sciences*, 1950.

[54] V. Srinivasan, P. Nuggehalli, C. F. Chiasserini, and R. R. Rao, “Cooperation in wireless ad hoc netwoks,” in *Proc. IEEE INFOCOM 2003*, pp. 808–817, 2003.

[55] W. Poundstone, *Prisoner’s Dilemma.* New York: Doubleday, February 1992.

[56] R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan, “Experiences applying game theory to system design,” in *ACM SIGCOMM Wksp Practice and Theory of Incentives and Game Theory In Networked Sys*, 2004.

[57] L. Huang, L. Li, L. Liu, H. Zhang, and L. Tang, “Stimulating cooperation in route discovery of ad hoc networks,” in *Proc. of the 3rd ACM Workshop on QoS and Security for Wireless and Mobile Networks (Q2SWinet’07)*, October 2007.

[58] K. Paul and D. Westhoff, “Context aware detection of selfish nodes in DSR based ad-hoc networks,” in *Proc. of IEEE Vehicular Technology Conference02*, 2002.

[59] OMNET++ Home Page, “http://www.omnetpp.org.” [accessed on May 20, 2009].

[60] INET Framework Home Page, “http://inet.omnetpp.org/.” [accessed on May 20, 2009].

[61] INETMANET Home Page, “http://webpersonal.uma.es/ AARIZAQ/.” [accessed on May 20, 2009].

[62] D. B. Johnson and D. A. Maltz, “Dynamic source roting in ad-hoc wireless networks,” technical report, Computer Science Department Carnegie Mellon University, 50000 Forbes Avenue Pittsburgh, PA, 1996.

[63] C.-K. Toh and E. Royer, “Review of current routing protocols for ad-hoc mobile wireless networks,” technical report, UCSB, 1999.

[64] S. R. Das, C. E. Perkins, and E. Royer, “Performance comparison of two on-demand routing protocols for ad-hoc networks,” in *INFOCOM 2000 Conference*, (Tel-Aviv, Israel), March 2000.

[65] A. Nasipuri, R. Castaneda, and S. Das, "Performance of multipath routing for on-demand protocols in mobile ad-hoc networks," *ACM/Kluwer Mobile Networks and Applications Journal*, vol. 6, no. 4, pp. 339–349, 2001.

[66] L. Wang, L. Zhang, Y. Shu, and M. Dong, "Multipath source routing in wireless ad-hoc networks," in *Canadian Conference on Electrical and Computer Engineering*, pp. 479–483, 2000.

[67] C. Tachtatzis, *Load Distribution and Energy Awareness in MANETs using Multipath Routing.* Phd thesis, University of Strathclyde, Department of Electrical & Electronic Engineering, April 2008.

[68] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: A survey," *IEEE Wireless Communication*, vol. 11, pp. 6–28, December 2004.

[69] J. R. Douceur, "The sybil attack," in *Proc. of the IPTPS02 Workshop*, (Cambridge, MA (USA)), March 2002.

[70] D. Nitnaware and A. Verma, "Performance evaluation of energy consumption of reactive protocols under self-similar traffic," *International Journal of Computer Science & Communication (IJCSC)*, vol. 1, pp. 67–71, February 2010.

[71] S. Abbas, M. Merabti, and D. Llewellyn-Jones, "Deterring whitewashing attacks in reputation based schemes for mobile ad hoc networks," in *Wireless Days (WD) IFIP*, pp. 1–6, October 2010.

[72] T. Anantvalee and J. Wu, "Reputation-based system for encouraging the cooperation of nodes in mobile ad hoc networks," in *Proc. of IEEE International Conference on Communication (ICC)*, (Glasgow), 2007.

[73] T. Anantvalee and J. Wu, *A Survey on Intrusion Detection in Mobile Ad Hoc Networks.* Springer, 2006.

[74] G. Angelou, *Mobile Ad Hoc Networks: From Wireless LANs to 4G Networks.* McGraw Hill, 2005.

[75] A. Babakhouya, Y. Challal, and A. Buouabdallah, "A simulation analysis of routing misbehavior in mobile ad hoc networks," in *The Second International*

Conference on Next Generation Mobile Applications, Services and Technologies NGMAST'08, pp. 592–597, 2008.

[76] K. A. A. Bakar and J. Irvine, "Contribution time-based selfish nodes detection scheme," in *The 11th Annual Postgraduate Sympossium on the Convergence of Telecommunications, Networking and Broadcasting (PGNET2010)*, (Liverpool, UK), pp. 271–275, June 2010.

[77] K. A. A. Bakar and J. Irvine, "A scheme for detecting selfish nodes in manets using OMNET++," in *The 6th International Conference on Wireless and Mobile Communications (IEEEE ICWMC 2010)*, (Valencia, Spain), pp. 410–414, September 2010.

[78] P. Bellavista and A. Corradi, "The handbook of mobile middleware." Auchback Publication, 2007.

[79] J. Broch, D. B. Johnson, and D. A. Maltz, "The dynamic source routing protocol for mobile ad hoc network," in *IETF*, February 2003. Internet Draft Version 08.

[80] S. Buchegger, *Coping with Misbehaviour in Mobile Ad Hoc Networks*. PhD thesis, Ecole Polytechnique Federale De Lausanne, 2004.

[81] S. Buchegger and J.-Y. L. Boudec, "Reputation systems for self-organized networks: Lesson learned," *IEEE Technology and Society Magazine, Special Issue on Limits of Cooperation in Wireless Communications, Towards Fourth Generation Wireless*, vol. 27, pp. 41–47, March 2008.

[82] S. Buchegger and J.-Y. L. Boudec, "Self-policing mobile ad hoc networks by reputation system," *IEEE Communication Magazine*, vol. 43, pp. 101–107, July 2005.

[83] S. Buchegger and J.-Y. L. Boudec, "A robust reputation system for peer-to-peer and mobile ad-hoc networks," in *Proc. of P2PEcon 2004*, Harvard University, Cambridge MA, U.S.A., June 2004.

[84] S. Buchegger and J.-Y. L. Boudec, "The effect of rumour spreading in reputation systems for mobile ad-hoc networks," in *Proc. of WiOpt03*, March 2003.

[85] M. Carvalho, "Security in mobile ad hoc networks," *Security & Privacy*, vol. 6, pp. 72–75, March-April 2008.

[86] Castalia Home Page, "http://castalia.npc.nicta.com.au." [accessed on May 20, 2009].

[87] J. Chen, X. Xu, and S. D. Bruda, "Combining data trust in reputation systems to boost p2p security," in *2nd International Conference on Future Computer and Communication (ICFCC)*, vol. 3, pp. 194–199, 2010.

[88] Cooperation in Wireless Network, "http://winet-coop.epfl.ch." [accessed on May 20, 2009].

[89] C. R. Davis, "A localized trust management scheme for ad hoc networks," in *The 3rd International Conference on Networking (ICN'04)*, pp. 671–675, March 2004 2004.

[90] T. Dean, *Network+ Guide to Networks*. Course Technology, 5 ed., 2010.

[91] D. Djenouri, O. Mahmoudi, M. Bouamama, D. Llewellyn-Jones, and M. Merabti, "On securing manet routing protocol against control packet dropping," in *The 4th IEEE International Conference on Pervasive Services (ICPS'2007)*, (Istanbul), pp. 100–108, July 2007.

[92] L. Eschenauer, J. S. Baras, and V. Gligor, "Distributed trust establishment in MANET's: swarm intelligence," in *Proc. of the Collaborative Technology Alliances (CTA) Communications & Network (C&N) Alliance-2003 Annual Symposium*, pp. 125–129, 2003.

[93] M. Frank, P. Martini, and M. Plaggemeier, "CineMA: cooperation enhancement in MANETs," in *Proc. of 29th Annual IEEE International Conference*, pp. 86–93, 2004.

[94] V. Garg, *Wireless Communication and Networking*. Elsevier, 2007.

[95] Glomosim Home Page, "http://pcl.cs.ucla.edu/projects/glomosim." [accessed on May 20, 2009].

[96] Q. He, D. Wu, and P. Khosla, "Sori: A secure and objective reputation-based incentive scheme for ad-hoc networks," in *WCNC 2004, Atlanta, GA*, March 2004.

[97] R. Hekmat, *Ad-hoc Networks: Fundamental Properties and Network Topologies.* Springer, 2006.

[98] J. Hu, "Cooperation in mobile ad hoc networks," 2005.

[99] H. Imai, *Wireless Communication Security.* Artech House, 2005.

[100] B. Ishibashi and R. Baotaba, "Topology and mobility considerations in mobile ad hoc networks," in *Ad Hoc Networks*, pp. 762–776, Nov 2005.

[101] N. Jain and D. P. Agrawal, "Current trends in wireless sensor network design," *International Journal of Distributed Sensor Networks*, vol. 1, pp. 101–122, 2005.

[102] W. K. Jr and L. Lazos, "Dealing with liars: Misbehavior identification via renyi-ulam games," in *5th International ICST Conference on Security and Privacy in Communication Networks (SecureComm 2009)*, vol. 19, pp. 207–227, Springer Berlin Heidelberg, September 2009.

[103] K. Kane and J. C. Browne, "Using uncertainty in reputation methods to enforce cooperation in ad hoc networks," in *Proc. of 5th ACM Workshops on Wireless Security*, (Los Angeles, CA, U.S.), pp. 105–113, 2006.

[104] V. Katiyar, N. Chand, and N. Chauhan, "Recent advances and future trends in wireless sensor networks," *International Journal of Applied Engineering Research*, vol. 1, pp. 330–342, 2010.

[105] H. J. Kim and J. M. Peha, "Detecting selfish behavior in a cooperative commons," in *3rd IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN 2008)*, pp. 1–12, October 2008.

[106] F. Li and J. Wu, "Mobility reduces uncertainty in manets," in *26th IEEE International Conference on Computer Communications (INFOCOM 2007)*, pp. 1946–1954, May 2007.

[107] S. Loyka and A. Kouki, "Using two ray multipath model for microwave link budget analysis," *IEEE Antennas and Propagation Magazine*, vol. 43, pp. 31–36, October 2001.

[108] M. Lu, F. Li, and J. Wu, "Incentive compatible cost and stability-based routing in ad hoc networks," in *Proc. of the 12th International Conference on Parallel and Distributed Systems*, pp. 495–500, 2006.

[109] C. Mallanda, A. Suri, V. Kunchakarra, S. S. Iyengar, R. Kannan, and A. Durresi, "Simulating wireless sensor networks with OMNET++." submitted to IEEE Computer, 2005.

[110] D. A. Maltz, "Demand routing in multi-hop wireless mobile ad hoc networks," May 2001.

[111] K. Mandalas, D. Flitzanis, G. F. Marias, and P. Georgiadis, "A survey of several cooperation enforcement schemes for manets," in *Proc. of IEEE International Symposium of Signal Processing and Information Technology*, 2005.

[112] G. F. Marias, P. Georgiadis, D. Flitzanis, and K. Mandalas, "Cooperation enforcement schemes for manets: A survey," in *Proc. of Wireless Communications and Mobile Computing*, pp. 319–332, 2006.

[113] P. McDermott-Wells, "What is bluetooth?," *Potentials, IEEE*, vol. 23, p. 33, December 2005.

[114] J. Michaels, "Military beefs up internet arsenal; terrorists expands use of computers." USA Today newspaper, March 2007.

[115] C. Michell, "Security for mobility." Institute of Electrical Engineering, 2004.

[116] F. Milan, J. J. Jaramillo, and R. Srikant, "Achieving cooperation in multi-hop wireless networks of selfish nodes," in *Proc. of Workshop on Game Theory for Communications and Networks*, 2006.

[117] N. Milanovic, M. Malek, A. Davidson, and V. Milutinovic, "Routing and security in mobile ad hoc networks," *Computers*, vol. 37, pp. 61–65, Feb 2004.

[118] H. Miranda and L. Rodrigues, “Friends and foes: Preventing selfishness in open mobile ad hoc networks,” in *23rd International Conference on Distributed Computing Systems Workshop*, pp. 440–445, 2003.

[119] A. Mishra, *Security and Quality of Service in Ad Hoc Wireless Networks*. Cambridge University Press, 2008.

[120] MiXiM Project Home Page, “http://mixim.sourceforge.net.” [accessed on May 20, 2009].

[121] C. S. R. Murthy and B. S. Manoj, *Ad Hoc Wireless Networks: Architectures and Protocols*. Prentice Hall, May 2004.

[122] Network Simulator 2 Home Page, “http://www.isi.edu/nsnam/ns.” [accessed on May 20, 2009].

[123] OPNET Home Page, “http://www.opnet.com.” [accessed on May 20, 2009].

[124] N. Oualha and Y. Roudier, “Cooperation incentive scheme,” September 2006.

[125] K. Paul and D. Westhoff, “Context aware inferencing to rate a selfish node in DSR based ad-hoc networks,” in *Proc. of IEEE Globecom Conference, Taipei, Taiwan*, 2002.

[126] C. Peng, “GSM and GPRS Security,” tech. rep., Helsinki University of Technology, 2001. http://www.tml.tkk.fi/Opinnot/Tik-110.501/2000/papers/peng.pdf.

[127] C. E. Perkins, E. M. Royer, and S. R. Das, “Ad hoc on-demand distance vector (aodv) routing (rfc3561),” in *The Internet Society*, 2003. Memo RFC 3561.

[128] T. Rappaport, *Wireless Communication: Principles and Practice*. NJ, USA: Prentice Hall, 1996.

[129] P. Resnick and R. Zeckhauser, “Trust among strangers in internet transaction: Empirical analysis of ebay’s reputation system,” 2002.

[130] Z. Safaei, M. Sabaei, and F. Torgheh, “An efficient reputation-based mechanism to enforce cooperation in manets,” in *International Conference on Application of Information and Communication Technologies (AICT 2009)*, 2009.

[131] G. Schafer, *Security in Fixed and Wireless Networks: An introduction to Securing Data Communication.* John Wiley & Sons, 2003.

[132] S. Sreepathi, V. Venigalla, and A. Lal, "A survey paper on security issues pertaining to ad-hoc networks," tech. rep., North Carolina State University, Raleigh, NC, USA, 2004.

[133] A. Srinivasan, J. Teitelbaum, H. Liang, J. Wu, and M. Cardei, "Reputation and trust-based systems for ad hoc and sensor networks," in *Algorithms and Protocols for Wireless Ad Hoc and Sensor Networks*, 2006.

[134] A. Srinivasan, J. Teitelbaum, and J. Wu, "DRBTS: distributed reputation-based beacon trust system," in *Proc. of Dependable, Autonomic and Secure Computing, 2nd IEEE International Symposium*, pp. 277–283, September 2006.

[135] W. Stallings, *Network Security Essentials: Applications and Standards.* Prentice Hall, 1999.

[136] K. Taneja and R. Patel, "Mobile ad hoc network: Challenges and future," in *Proc. of National Conference on Challenges & Opportunities in Information Technology (COIT-2007)*, March 2007.

[137] The OverSim P2P Simulator Home Page, "http://www.oversim.org." [accessed on May 20, 2009].

[138] G. Theodorakopoulos and J. S. Baras, "Trust evaluation in ad-hoc networks," in *3rd ACM Workshop on Wireless Security*, pp. 1–10, 2004.

[139] VoIPTool Home Page, "http://www.tkn.tu-berlin.de/research/omnetvoiptool." [accessed on May 20, 2009].

[140] B. Wu, J. Chen, and M. Cardei, *A Survey on Attacks and Countermeasures in Mobile Ad Hoc Networks.* Springer, 2007.

[141] G. Xiapeng and C. Wei, "A novel gray hole attack detection scheme for mobile ad-hoc networks," in *IFIP International Conference on Network and Parallel Computing*, pp. 209–214, September 2007.

[142] P. W. Yau and C. J. Mitchell, "Reputation methods for routing security for mobile ad hoc networks," in *Mobile Future and Symposium on Trends in Communications (SympoTIC '03)*, 2003.

[143] Y. Yoo and D. P. Agrawal, "Why does it pay to be selfish in a manet?," *IEEE Wireless Communications*, vol. 13, pp. 87–97, December 2006.

[144] Y. Zhang, *Handbook of research on wireless security.* Idea Group Inc, 2008.

[145] Y. Zhang, J. Luo, and H. Hu, "Wireless mesh networking: architectures, protocols and standards," in *Technology and Engineering*, pp. 188–189, Auerbach Publications, 2006.

[146] "IEEE standard 802.16-2004," October 2004.