A Cross-Layer and Multi-Metric Routing Decision Making Framework for MANETs

A thesis submitted to the University of Manchester for the degree of Doctor of Philosophy in the Faculty of Engineering and Physical Sciences

2013

By
Kitisak Osathanunkul
School of Computer Science
Contents

Abstract 17
Declaration 18
Copyright 19
Acknowledgements 20

1. Introduction 21
   1.1 Background 21
   1.2 Research Hypothesis and Challenges 24
   1.3 Research Aim and Objectives 25
   1.4 Novel Contributions and Publications 26
   1.5 Thesis Structure 28

2. MANET Routing 30
   2.1 Chapter Introduction 30
   2.2 OSI Model 30
   2.3 MANET Routing 33
       2.3.1 MANET Routing Introduction 33
       2.3.2 Routing Metrics 36
       2.3.3 MANET Routing Protocol 38
           2.3.3.1 Destination Sequence Distance Vector (DSDV) Routing Protocol 38
           2.3.3.2 Ad hoc On-demand Distance Vector (AODV) Routing Protocol 39
           2.3.3.3 Expected Transmission Count (ETX) 40
           2.3.3.4 A Trust-based Routing Protocol (TRP) 45
2.4 Security Threats and Attacks

2.4.1 Eavesdropping Attacks
2.4.2 Message Dropping Attacks
2.4.3 Message Modification and Fabrication Attacks
2.4.4 Delay Attacks (Jellyfish Attacks)
2.4.5 Replay Attacks
2.4.6 Impersonation Attacks
2.4.7 Flooding Attacks
2.4.8 Black Hole Attacks
2.4.9 Selfish Nodes

2.5 The Best Way Forward

2.6 Chapter Summary

3. A Countermeasure to Black Hole Attacks in Mobile Ad hoc Networks

3.1 Chapter Introduction

3.2 ETX Security Analysis

3.2.1 Case 1: Advertising a Falsified forward delivery ratio
3.2.2 Case 2: Modifying BROADCAST_RATE
3.2.3 Case 3: Impersonation Attacks (Active Selfish Attacks)
3.2.4 ETX Security Analysis Summary

3.3 The Secure ETX (SETX) Protocol

3.3.1 Protocol Overview
3.3.2 Detailed Protocol Description

3.3.3 Probe Message Protection using Cryptographic Techniques (An Optional Solution)

3.3.3.1 Digital Signature Technique
3.3.3.2 Keyed Hash Function Technique
3.4 Security Analysis of the SETX Protocol

3.4.1 Security Analysis against Security Attacks on ETX

3.4.1.1 Attacks on ETX Case 1: Advertising a falsified forward delivery ratio

3.4.1.2 Attacks on ETX Case 2: Modifying BROADCAST_RATE

3.4.1.3 Attacks on ETX Case 3: Impersonation Attacks (Active Selfish Attacks)

3.4.2 Security Analysis against Security Attacks on SETX

3.4.2.1 Attacks on SETX Case 1: Probe Message Guessing Attacks

3.4.2.2 Attacks on SETX Case 2: Cooperative Black Hole Attacks

3.4.2.3 Attacks on SETX Case 3: Impersonation Attacks (Black Hole Attacks)

3.4.3 Discussions of the Security Analysis

3.5 Comparing the SETX Protocol with Related Works

3.5.1 Related Works

3.5.2 Comparing SETX against Shila's Algorithm

3.5.3 Summary

3.6 SETX Protocol Performance Evaluation

3.6.1 Simulation Modelling

3.6.1.1 Routing Models

3.6.1.2 Mobile Node Models

3.6.1.3 Malicious Node Model

3.6.2 Simulation Parameters

3.6.3 Simulation Results

3.6.3.1 Scenario 3.1: No Intermediate Node

3.6.3.2 Scenario 3.2: With Intermediate Node
3.6.3.3 Scenario 3.3: With Moving Intermediate Node

3.6.3.4 Scenario 3.4: With a Malicious Node outside the Best Route

3.6.3.5 Scenario 3.5: With a Malicious Node inside the Best Route

3.7 SETX Limitations

3.8 Chapter Summary

4. A Flexible Routing Decision (FRD) framework for MANETs

4.1 Chapter Introduction

4.2 Problem Statement

4.3 A Literature Review: Routing Algorithms for Routing Decision Making
   4.3.1 Single Routing Metric Type Based Routing Algorithms
   4.3.2 Multiple Routing Metric Type Based Routing Algorithms
      4.3.2.1 The Rank Order Method
      4.3.2.2 The Threshold (or ε-Constraint) Method

4.4 Multi-Criteria Decision Making (MCDM) Technique
   4.4.1 MCDM in MANET Routings
   4.4.2 The Weighted Sum Model (WSM) Technique
   4.4.3 The Weighted Product Model (WPM) Technique
   4.4.4 The Analytic Hierarchy Process (AHP) Technique
   4.4.5 The Revised Analytic Hierarchy Process (RAHP) Technique
   4.4.6 Further Discussions

4.5 FRD Framework
   4.5.1 FRD Overview
   4.5.2 Detailed FRD framework Description
   4.5.3 FRD Discussions

4.6 A Simulation Study
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.6.1</td>
<td>Investigation of Scenario 4.1</td>
<td>154</td>
</tr>
<tr>
<td>4.6.1.1</td>
<td>Scenario Description</td>
<td>154</td>
</tr>
<tr>
<td>4.6.1.2</td>
<td>Route Selections</td>
<td>158</td>
</tr>
<tr>
<td>4.6.1.3</td>
<td>Simulation Setup</td>
<td>163</td>
</tr>
<tr>
<td>4.6.1.4</td>
<td>Simulation Results</td>
<td>165</td>
</tr>
<tr>
<td>4.6.2</td>
<td>Investigation of Scenario 4.2</td>
<td>172</td>
</tr>
<tr>
<td>4.6.2.1</td>
<td>Scenario Description</td>
<td>172</td>
</tr>
<tr>
<td>4.6.2.2</td>
<td>Route Selections</td>
<td>173</td>
</tr>
<tr>
<td>4.6.2.3</td>
<td>Simulation Results</td>
<td>177</td>
</tr>
<tr>
<td>4.6.2.4</td>
<td>Discussion of Simulation Results</td>
<td>184</td>
</tr>
<tr>
<td>4.7</td>
<td>Integration of the FRD Framework with the Real Network System</td>
<td>185</td>
</tr>
<tr>
<td>4.8</td>
<td>Chapter Summary</td>
<td>190</td>
</tr>
<tr>
<td>5.</td>
<td>Conclusions and Future Work</td>
<td>192</td>
</tr>
<tr>
<td>5.1</td>
<td>Thesis Conclusions</td>
<td>192</td>
</tr>
<tr>
<td>5.2</td>
<td>Contributions</td>
<td>194</td>
</tr>
<tr>
<td>5.3</td>
<td>Future Work</td>
<td>195</td>
</tr>
</tbody>
</table>

Bibliography

Appendix A. Probabilities of Guessing n Probe Message 208
Appendix B. NS2 Configurations 211
Appendix C. WPM Algorithm for FRD Decisions 214
Appendix D. Full Simulation Results for SETX protocols 219
Appendix E. Full Simulation Results for FRD 234

Word count: 62,594
List of Tables

Table 3.1 The Comparison of ETX and SETX Protocols 69
Table 3.2 The Comparison List of Attacks on ETX and SETX 98
Table 3.3 The Comparison of Different Security Solutions against Black Hole Attacks 105
Table 3.4 Simulation Parameter Settings 110
Table 4.1 The Example of a Routing Table that uses of the Tank Order Method in Multi-Criteria Routing Decision 133
Table 4.2 The Example of a Table of Routing Protocols that uses the Threshold Method in Multi-Criteria Routing Decision 138
Table 4.3 The Decision Table of Example Scenario 1 when the WSM Technique is used 139
Table 4.4 A Decision Table of the Example Scenario 1 when using the AHP Technique 142
Table 4.5 A Decision Table of the Example Scenario 2 when using the AHP Technique 143
Table 4.6 A Decision Table of the Example Scenario 3 when using the AHP Technique 143
Table 4.7 A Revised Decision Table of the Example Scenario 3 when using the AHP Technique 144
Table 4.8 Comparison Table between WSM, WPM, AHP and RAHP 145
Table 4.9 Deciding a Weight for each RRMT of all RRMT sets 150
Table 4.10 RRMTW Table for Scenario 4.1 157
Table 4.11 The Delay Metric Values of all routing candidates 159
Table 4.12 A Decision Table of RAHP for Delay Sensitive Data Type in Scenario 4.1 162
Table 4.13 A Decision Table of RAHP for Reliability Sensitive Data Type in Scenario 4.1 162
Table 4.14  A Decision Table of RAHP for General Application Data Type in Scenario 4.1 162
Table 4.15  The parameters used in the simulation 164
Table 4.16  A Decision Table of RAHP for Delay Sensitive Data Type in Scenario 4.2 176
Table 4.17  A Decision Table of RAHP for Reliability Sensitive Data Type in Scenario 4.2 176
Table 4.18  A Decision Table of RAHP for General Application Data Type in Scenario 4.2 176
Table 4.19  Summary of Routing Decisions 177
Table 4.20  Summary of the simulation results reflecting on the requirements of the application 184
Table 5.1  The RRMTW Table of the Example Scenario 197
Table 5.2  A Decision Table for General Application Data Type when using RAHP technique 197
Table 5.3  List of Metric Types and their Value Ranges used by the FRD Framework 198
List of Figures

Figure 1.1  MANET of 4 nodes  22
Figure 2.1  OSI Model with 7 Layers  31
Figure 2.2  MANET Routing Operation Cycle Model  33
Figure 2.3  The Broadcasting of a Route Request Packet  34
Figure 2.4  The Replying of an Acknowledgement Packet  34
Figure 2.5  Routes Found by the Source Node S  35
Figure 2.6  The difference between forward delivery ratio and reverse delivery ratio when focussing at different nodes  41
Figure 2.7  The ETX Protocol Procedures  42
Figure 2.8  The Black Hole Attack Model  51
Figure 2.9  RREQ modification  52
Figure 2.10  RREP modification  53
Figure 3.1  Advertising a falsified forward delivery ratio – Step 1  58
Figure 3.2  Advertising a falsified forward delivery ratio – Step 2  59
Figure 3.3  Advertising a falsified forward delivery ratio – Step 3  59
Figure 3.4  Modifying $BROADCAST\_RATE$ – Step 1  62
Figure 3.5  Modifying $BROADCAST\_RATE$ – Step 2  62
Figure 3.6  Modifying $BROADCAST\_RATE$ – Step 3  63
Figure 3.7  Impersonation Attacks – Step 1  65
Figure 3.8  Impersonation Attacks – Step 2  66
Figure 3.9  Impersonation Attacks – Step 3  66
Figure 3.10  Impersonation Attacks – Step 4  67
Figure 3.11  Categorisation of Security attacks on the ETX protocol  68
Figure 3.12  The use of APBs and RPBs in the SETX protocol  71
Figure 3.13  Procedures of the SETX protocol  72
Figure 3.14  SETX Procedure Step 1, Generating Probe Packets  72
Figure 3.15  Probe Packet Format  73
Figure 3.16  SETX Procedure Step 2, Advertising Probe Packets  76
Figure 3.17  SETX Procedure Step 3, Receiving Probe Packets  77
Figure 3.18  SETX Procedure Step 4, Storing Probe Message  78
Figure 3.19  SETX Procedure Step 5, Exchanging RPBs  79
Figure 3.20  Formats of RREQ and RREP packets used in SETX  79
Figure 3.21  Probe Verification Algorithm  80
Figure 3.22  Generation and Verification of Digital Signature on a Probe Message  83
Figure 3.23  Initiator $I$ generates a probe packet for neighbouring node $A$ and node $B$ using a secret key technique  84
Figure 3.24  Node A verifies a probe message ($pm_I$)  85
Figure 3.25  Probe Message Guessing Attacks – Step 1  89
Figure 3.26  Probe Message Guessing Attacks – Step 2  89
Figure 3.27  Probe Message Guessing Attacks – Step 3  90
Figure 3.28  Cooperative Black Hole Attacks – Step 1  91
Figure 3.29  Cooperative Black Hole Attacks – Step 2  92
Figure 3.30  Cooperative Black Hole Attacks – Step 3  92
Figure 3.31  Cooperative Black Hole Attacks – Step 4  93
Figure 3.32  Cooperative Black Hole Attacks – Step 5  93
Figure 3.33  Cooperative Black Hole nodes $M_1$ to $M_4$ receiving only 1 probe message each  94
Figure 3.34  Cooperative Black Hole nodes $M_1$ to $M_4$ exchanging their probe messages with each other  94
Figure 3.35  Impersonation Attacks on SETX – Step 1  95
Figure 3.36  Impersonation Attacks on SETX – Step 2  96
Figure 3.37  Impersonation Attacks on SETX – Step 3
Figure 3.38  Impersonation Attacks on SETX – Step 4
Figure 3.39  Impersonation Attacks on SETX – Step 5
Figure 3.40  The example of Routing Information Verification Scheme
Figure 3.41  Network Topology of Scenario 3.1
Figure 3.42  Average Throughputs in Scenario 3.1 with the data rate of 1 packet/second
Figure 3.43  Average Throughputs in Scenario 3.1 with the data rate of 10 packets/second
Figure 3.44  Average Throughputs in Scenario 3.1 with the data rate of 100 packets/second
Figure 3.45  Simulation Durations in Scenario 3.1 with the data rate of 1 packet/second
Figure 3.46  Simulation Durations in Scenario 3.1 with the data rate of 100 packets/second
Figure 3.47  Control Packet Counts in Scenario 3.1 with the data rate of 1 packet/second
Figure 3.48  Control Packet Counts in Scenario 3.1 with the data rate of 10 packets/second
Figure 3.49  Control Packet Rates in Scenario 3.1 with the data rate of 1 packet/second
Figure 3.50  Control Packet Rates in Scenario 3.1 with the data rate of 10 packets/second
Figure 3.51  Packet Delivery Ratios in Scenario 3.1 with the data rate of 1 packet/second
Figure 3.52  Network Topology of Scenario 3.2
Figure 3.53  Average Throughputs in Scenario 3.2 with the data rate of 100 packets/second
Figure 3.54  Control Packet Rates in Scenario 3.2 with the data rate of 1 packet/second
Figure 3.55  Network Topology of Scenario 3.3
Figure 3.56  Average Throughputs in Scenario 3.3 with the data rate of 100 packets/second  
Figure 3.57  Average Throughputs in Scenario 3.3 with the data rate of 1,000 packets/second  
Figure 3.58  Average Throughputs in Scenario 3.2 with the data rate of 1,000 packets/second  
Figure 3.59  Packet Delivery Ratios in Scenario 3.3 with the data rate of 1,000 packets/second  
Figure 3.60  Packet Delivery Ratios in Scenario 3.2 with the data rate of 1,000 packets/second  
Figure 3.61  Network Topology of Scenario 3.4  
Figure 3.62  Average Throughputs in Scenario 3.4 with the data rate of 100 packets/second  
Figure 3.63  Packet Delivery Ratios in Scenario 3.4 with the data rate of 100 packets/second  
Figure 3.64  Network Topology of Scenario 3.5  
Figure 3.65  Packet Delivery Ratios in Scenario 3.5 with the data rate of 100 packets/second  
Figure 3.66  Average Throughputs in Scenario 3.5 with the data rate of 100 packets/second  
Figure 4.1  The components of the FRD framework  
Figure 4.2  The example of using the FRD framework for Application Data Type A and Application Data Type B  
Figure 4.3  Procedures of the FRD framework  
Figure 4.4  RRMT Domain of a Node Running Application Data Type A and Application Data Type B  
Figure 4.5  Format of RREQ packets when using the FRD framework  
Figure 4.6  Format of RREP packets when using the FRD framework  
Figure 4.7  The topology of the network understudy, Scenario 4.1  
Figure 4.8  Average Throughputs for Reliability Sensitive Data Type in Scenario 4.1
Figure 4.9  Average Throughputs for General Application Data Type in Scenario 4.1  

Figure 4.10  Average Throughputs for Delay Sensitive Data Type in Scenario 4.1  

Figure 4.11  Route Breaking Times for General Application Data Type in Scenario 4.1  

Figure 4.12  Route Breaking Time in Routing Candidate A with Different Signalling Rates in Scenario 4.1.1  

Figure 4.13  Remaining Battery Levels of Routing Candidate A with Different Signalling Data Rates in Scenario 4.1.1  

Figure 4.14  Packet Delivery Ratios for Reliability Sensitive Data Type in Scenario 4.1  

Figure 4.15  Packet Delivery Ratios for General Application Data Type in Scenario 4.1  

Figure 4.16  Packet Delivery Ratios for Delay Sensitive Data Type in Scenario 4.1  

Figure 4.17  The topology of the network understudy, Scenario 4.2  

Figure 4.18  Average Throughputs for Delay Sensitive Data Type in Scenario 4.2  

Figure 4.19  Packet Delivery Ratios for Delay Sensitive Data Type in Scenario 4.2  

Figure 4.20  Packet Delivery Ratios for Reliability Sensitive Data Type in Scenario 4.2  

Figure 4.21  Simulation Durations for Reliability Sensitive Data Type in Scenario 4.2  

Figure 4.22  Average Throughputs for General Application Data Type in Scenario 4.2  

Figure 4.23  Packet Delivery Ratios for General Application Data Type in Scenario 4.2  

Figure 4.24  Simulation Durations for General Application Data Type in Scenario 4.2  

Figure 4.25  The example of the high-level requirements of the application called Skype
Figure 4.26  The example of IPv4 Protocol Suite with the FRD Framework  

Figure 4.27  The example of the FRD Framework Activation  

Figure 4.28  The example of MCDM Technique Settings for General Applications  

Figure 4.29  The example of RRMT and RRMTW Settings for General Applications  

Figure 5.1  The example of normalising ETX metric values
## Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AODV</td>
<td>Ad hoc On demand Distance Vector Protocol</td>
</tr>
<tr>
<td>AHP</td>
<td>Analytic Hierarchy Process</td>
</tr>
<tr>
<td>AHPS</td>
<td>Analytic Hierarchy Process Score</td>
</tr>
<tr>
<td>APB</td>
<td>Advertised Probe Buffer</td>
</tr>
<tr>
<td>CBR</td>
<td>Constant Bit Rate</td>
</tr>
<tr>
<td>DSDV</td>
<td>Destination Sequence Distance Vector Protocol</td>
</tr>
<tr>
<td>FRD</td>
<td>Flexible Routing Decision</td>
</tr>
<tr>
<td>RAHP</td>
<td>Revised Analytic Hierarchy Process</td>
</tr>
<tr>
<td>RAHPS</td>
<td>Revised Analytic Hierarchy Process Score</td>
</tr>
<tr>
<td>RC</td>
<td>Routing Candidate</td>
</tr>
<tr>
<td>RIP</td>
<td>Routing Information Protocol</td>
</tr>
<tr>
<td>RPB</td>
<td>Received Probe Buffer</td>
</tr>
<tr>
<td>RREP</td>
<td>Route Reply Packet</td>
</tr>
<tr>
<td>RREQ</td>
<td>Route Request Packet</td>
</tr>
<tr>
<td>RRMT</td>
<td>Relevant Routing Metric Type</td>
</tr>
<tr>
<td>RRMTW</td>
<td>Relevant Routing Metric Type Weight</td>
</tr>
<tr>
<td>RRMV</td>
<td>Relevant Routing Metric Value</td>
</tr>
<tr>
<td>RTT</td>
<td>Round Trip Time</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>ETX</td>
<td>Expected Transmission Count</td>
</tr>
<tr>
<td>MANET</td>
<td>Mobile Ad hoc Network</td>
</tr>
<tr>
<td>MCDM</td>
<td>Multi-Criteria Decision Making</td>
</tr>
<tr>
<td>NS2</td>
<td>Network Simulator 2</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>SETX</td>
<td>Secure Expected Transmission Count</td>
</tr>
<tr>
<td>TRP</td>
<td>Trust-based Routing Protocol</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>WPM</td>
<td>Weighted Product Method</td>
</tr>
<tr>
<td>WPMR</td>
<td>Weighted Product Method Ratio</td>
</tr>
<tr>
<td>WSM</td>
<td>Weighted Sum Method</td>
</tr>
<tr>
<td>WSMS</td>
<td>Weighted Sum Method Score</td>
</tr>
</tbody>
</table>
Abstract

Mobile Ad hoc Networks (MANETs) are re-emerging as a popular networking facility for wireless device users. A growing number of diversified applications are now accessible via wireless devices. The different applications may have different Quality of Service (QoS) requirements, which may better be satisfied by using different routing methods or metric types. Existing ad hoc network routing solutions do not consider various application-level requirements when making a routing decision. They typically make routing decisions based upon limited information acquired at the network layer. Most of the existing routing protocols make use of a single routing metric. Using a single metric type and/or information, only acquired at the network layer may not be able to accommodate different QoS requirements, imposed by diversified user-level applications or application-level data types.

The aim of this thesis is to design an efficient routing function for ad hoc networks while at the same time satisfying users’ and/or applications’ QoS and security requirements. To achieve this, the thesis investigates and specifies routing requirements that could best support application-level QoS and security requirements in MANETs. It also investigates and critically analyses the state of the art in MANET routing, and the mechanisms used for protecting the routing functions. To overcome the weaknesses and advance the state of the art in MANET routing, this thesis proposes two major solutions. The first solution is the Secure ETX (SETX) routing protocol. It is a secure routing solution that can provide routing functions efficiently in malicious MANET environment. The SETX protocol provides a security mechanism to counter black hole attacks in MANETs on the ETX metric acquisition process. Simulation studies have been carried out and discussed in the thesis. Simulation results show that the SETX protocol can provide a marked improvement in network performances in the presence of black hole attacks, and it can do so with a negligible level of additional overhead.

The second solution is a novel routing decision making called the Flexible Routing Decision (FRD) framework. The FRD framework supports routing decision making by using multiple metric types (i.e. multi-criteria routing decision making) and uses a cross-layer approach to support application-level QoS requirements. This allows users to use different routing metrics types, making the most appropriate routing decision for a given application. To accommodate the diversified application-level QoS requirements, multiple routing metric types have been identified and interpreted in the FRD framework design. The FRD framework has overcome some weaknesses exhibited by existing single metric routing decision making, used in MANETs. The performance of a routing decision making of FRD is also evaluated using NS2 simulation package. Simulation results demonstrate that the FRD framework outperforms the existing routing decision making methods.
Declaration

No portion of the work referred to in the thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning
Copyright

Copyright in text of this thesis rests with the Author. Copies (by any process) either in full, or of extracts, may be made only in accordance with instructions given by the Author and lodged in the John Ryland’s University Library of Manchester. Details may be obtained from the Librarian. This page must form part of any such copies made. Further copies (by any process) of copies made in accordance with such instructions may not be made without the permission (in writing) of the Author.

The ownership of any intellectual property rights which may be described in this thesis is vested in the University of Manchester, subject to any prior agreement to the contrary, and may not be made available for use by third parties without the written permission of the University, which will prescribe the terms and conditions of any such agreement.

Further information on the conditions under which disclosures and exploitation may take place is available from the Head of the School of Computer Science.
Acknowledgement

I would like to express my deep gratitude to my supervisor Dr. Ning Zhang, to whom I am very much obliged for her time, guidance and valuable advice throughout this research. Her support was essential from the moment I started my PhD to the last minute suggestion before submitting the thesis.

My deepest gratitude and appreciation to my family: My Mum, Maslin, Rossarin, Khukrit and Chin, for their help and support in difficult time. Your love, understanding and patience sustained me through to the end of my PhD.

I cannot thank my parents enough for their love, their support, and sacrifice throughout my whole life. Over the years they have continued to support and encourage me. I would like to thank them for everything.

I would like to thank those closest to me, whose presence helped make the completion of my thesis. These are Meng Chang Liu, Nam Shik Han and Mao Lu, my best friend during my PhD study. I am also thankful to Kwanchai Eurviriyakul, Wai Lin Chan, Joy Bose and Vegunta Sarat Chandra, who shared my happiness and made me happy during the difficult time. Last but not least, Cathleen Liu, thanks for the love, patience, understanding, and believing that one day this thesis will be completed.
Chapter 1

Introduction

1.1 Background

In wireless networking, there are two modes for establishing a wireless connection, infrastructure mode and ad hoc mode. Infrastructure mode requires the use of a wireless access point. Wireless enabled devices such as laptops, Personal Digital Assistants (PDAs) and mobile phones typically use this mode to connect to wireless networks which are often connected to a backbone or the internet through a router or an access point. Since the devices in this mode access a network via an access point, their connectivity is limited within the coverage areas of the access point.

In the ad hoc mode, on the other hand, wireless devices communicate with each other spontaneously, and they do not require the use of access point or a gateway like the case in infrastructure mode. The resulting network is also called a Mobile Ad Hoc Network (MANET) where mobile devices (mobile nodes) can communicate directly with each other when they are within wireless transmission range of each other. In addition, MANETs also support communications between nodes that are not directly connected. The communications are routed through other nodes in the network. Using the scenario shown in Figure 1.1 as an example, there are 4 nodes in the network. Node A is connected to node B. Node B is connected to Node A and C. Now node A wants to communicate with node D. Node B and C can help to relay packets from node A to node D. This is called a multi-hop wireless communication. In such networks, there are no dedicated routers; communicating nodes act as routers helping each other to forward traffic.

Owing to the node mobility, network topologies in MANETs change frequently and can be unpredictable. Traditional routing protocols designed for infrastructure wireless networks, such as distance-vector and link-state routing protocols, are not readily suited to MANETs [TOH02]. They are not designed to accommodate the dynamic and self-configuring nature of MANETs. Also routing functions in MANETs are performed by mobile devices. These devices have more limited resources than dedicated routers. Therefore, routing protocols
designed for MANET environments [HAA02, JOH96, PER94, PER97, RFC3626, SHR96] should be computationally efficient.

Furthermore, owing to the increasing affordability of wireless devices, MANETs have found a variety of applications in our daily lives, and in a variety of forms, e.g. from Personal Area Networks (PAN) to Vehicular Ad hoc Networks (VANETs). Since mobile devices are becoming more powerful in terms of storage and computational capability, applications that are available on wired devices are increasingly run on wireless devices. Deployment of these diversified of applications in a MANET may warrant provisioning of Quality of Service (QoS). However, provisioning QoS can be difficult to achieve in MANET. Two major difficulties that are focused on in this thesis are a diversity of applications and security attacks.

Firstly, the diversity of applications makes a routing decision in MANETs become difficult. Different applications may generate different data types, which in turn, may impose different routing requirements which may be better served with the use of different routing criteria. Existing MANET routing protocols do not consider various application-level requirements (e.g. less delay, high reliability and long availability) when making a routing decision. They typically make a routing decision based upon limited information acquired at the network layer (e.g. hop count, this will be described in Section 2.3.2).
of the existing routing protocols make use of a single routing metric. For example, the Ad hoc On demand Distance Vector (AODV) routing protocol [PER97] uses hop count as the underlying routing metric to select the shortest route among those available. The route selected using this metric may be appropriate for forwarding general data. However, it may not be the best route for reliability-sensitive applications, for example, emergency services. The shortest route may not be the most reliable route. Therefore using a single metric type and/or information acquired at the network layer only, may not be able to accommodate different QoS requirements imposed by diversified user-level applications or application-level data types.

Secondly, security attacks also pose a considerable threat to QoS provisioning in MANETs. The limited resources of mobile nodes may encourage nodes to be selfish in serving other nodes in order to preserve their own resources. This malicious intention leads to security attacks which can delay or disrupt the network operation. An example of this attack is black hole attacks. Black hole nodes may drop a packet (e.g. data packets or control packets) in order to preserve their resources or to disrupt the network operation. In addition, the implication of this attack (where the packet is dropped or the communication link is cut down) is similar to general failures because of MANET characteristics. The examples of general failures in MANETs include link breakage (resulting from nodes roaming out of wireless range) and battery depletion (i.e. the limited battery life of mobile nodes and expensive nature of wireless communication). The outcome of both security attacks and general failures is that the MANET connections are dynamic and unstable. This makes QoS provisioning a challenging task.

Research activities in the domain of QoS provisioning and in securing routing procedures in MANETs are very much separated. The works on QoS tend to ignore security problems and their impacts on QoS [RIS09, SAR06, ZHA10]. On the other hand, works in the security field rarely considers QoS issues [AWE03, HU02, HU05, LIN06, RAM07, ZAP05]. However, these two areas are closely related. Security provisioning introduces overhead and consumes bandwidth thus depleting QoS, and on the other hand without considering security (such as the implication of black hole attacks), QoS will suffer.
1.2 Research Hypothesis and Challenges

The focus of this research is to design a routing solution using a cross-layer and multi-metric approach. Different applications, depending on their data types and/or user-level preferences, may have different QoS and security requirements. Different QoS and security requirements may better be satisfied by different routing metrics at the network layer. Some may require the use of more than one metric. It may be desirable to have a solution that could map a given set of QoS requirements of an application to an appropriate set of metrics, and select the best route using the values of these metrics for the application. In other words, there is a need to identify a set of routing requirements of application-level data types by which one or more routing metric types could be selected. This set of criteria is typically defined by applications, but their values should be conveyed onto the routing decision engine run at the network layer. It is hypothesised that such a cross-layer and multi-metric routing solution could better satisfy application-level QoS requirements in terms of reliability, security, and performances.

To verify this hypothesis, it is necessary to address a number of challenging issues including:
- How to map application-level requirements onto network layer routing criteria or metrics? The designed method should be able to accommodate a number of application-level requirements and multiple routing metric types, and should be flexible enough to support the addition of a new application-level requirement or a routing metric type, and the deletion of an obsolete one.
- For a different routing criterion, usually a different routing metric type\(^1\) is used, which is typically implemented by a specific routing algorithm. So, how to evaluate multiple routing metric types that are implemented by dissimilar routing algorithms?
- How to make a routing decision if an application has multiple and possibly conflicting requirements? Should multiple routing metrics be considered?
- It can be difficult to decide which security mechanism should be chosen for a routing instance. Different security mechanisms provide different security properties, addressing different forms of threats and attacks, e.g. impersonation

\(^1\) The example of routing metric types are hop count, trust metric and remaining battery level metric.
attacks, attacks on confidentiality, on availability and integrity. The selection of a security mechanism should also consider factors including the computational cost of a mechanism, current security threat level in the environment, reliability of the link, a trustworthiness of the neighbouring node/route, in addition to application-level requirements.

- How to minimise additional overheads when designing the novel routing solution? Excessive computational or communication overheads introduced by having the additional capability offered by the novel solution will offset the benefit brought about by the solution. Therefore it is very important to minimise the overheads incurred by having the additional functionality.

- The design should take a modular approach such that any change made to one or any of the functional blocks in the novel solution should not cause any change to another functional block. This requirement is necessary to ensure that the solution could be applied in different application contexts with minimum modifications, and can easily be extended to satisfy different user requirements. For example, different applications may have different security and performance requirements, some applications may have more fine-grained requirements than others, and they may use different attributes or metrics. The solution should be flexible enough to cater for these diversified requirements and scenarios. Furthermore, new routing ideas and algorithms may emerge in the future. Should this happen, our solution should be easily extendible to embed emerging technologies.

1.3 Research Aim and Objectives

The aim of this research is to find an approach to provide an efficient routing function for ad hoc networks while at the same time satisfying users’ and/or applications’ QoS and security requirements. In order to achieve this aim, the objectives of the research are as follows:

1. To thoroughly understand and specify routing requirements that could best support application-level QoS and security requirements in MANETs.

2. To investigate and critically analyse the state-of-the-art in MANET routing, and the mechanisms used for protecting the routing functions.
3. To examine ways in which the weaknesses of the existing solutions might be addressed.
4. To overcome the weaknesses and advance the state of the art by designing a secure routing solution that can provide routing functions efficiently in a malicious MANET environment.
5. To perform security analysis and performance evaluation of the designed solution in order to study its effectiveness and efficacy.

1.4 Novel Contributions and Publications

The main contributions of this thesis are twofold. The first contribution is the design and simulation evaluation of the Secure ETX (SETX) protocol. The SETX protocol is designed to acquire the value of the ETX routing metric (introduced in Expected Transmission Count protocol [DEC03]) more reliably in an adversarial MANET environment. The design of the ETX metric is aimed to find a route which can provide the least loss rate in a homogeneous sensor network. It estimates the loss rate of each available route by measuring packet loss ratios of each direction of a wireless link. Simulation results [DEC03] have shown that the ETX metric approach is more effective in terms of finding a better route (i.e. one with a lower loss rate) than the popular minimum hop count approach, particularly for routes with two or more hops. However, the original design of the ETX approach does not take into account black hole attacks. Black hole nodes behave maliciously by fabricating a routing metric value in order to lure traffic to pass through them. They then drop data packets to disrupt the network operations. The value of the ETX metric can be significantly distorted in the presence of this attack. As a result, routes selected based on this metric value may not be optimal. The SETX protocol is designed to thwart black hole attacks on a route discovery process and to ensure that the ETX metric values that are acquired from neighbouring nodes are more reliable, thus making the routing process more reliable and network communication more efficient.

The second contribution is the design and simulation evaluation of the Flexible Routing Decision (FRD) framework. It is designed to support application-level QoS and security requirements by enabling routing decision making at the network layer. The framework supports the use of multiple routing metrics types in one platform. The number of routing
metric types that can be supported is not limited to any number. It allows devices to weigh, rank and use different types of routing metrics to select different routes on an individual application basis. This aims to find the most suited route to their application requirements.

The strengths of the FRD framework are three-fold.

(1) It allows the selections of routes at the network layer based upon application-level requirements. So the traffic generated by an application can be forwarded along the route that is chosen based upon application-level requirements. The application-level requirements may be determined by application types, data types generated by the applications, and/or users’ security and performance preferences.

(2) In addition, to better satisfy application-level QoS and security requirements, FRD is also designed to better utilise network bandwidth resources, leading to a better balanced traffic distribution across the entire network. This, in turn, can help to further reduce routing delays and improve routing efficiency. The method can evaluate the composite effect of multiple routing metric values and feed this effect into a route selection/discovery process. This additional capability can help to better satisfy the application-level QoS and security requirements.

(3) Flexibility and compatibility of the framework is one of the design strengths. The FRD framework allows nodes to negotiate the routing metric types that they are going to use before the routing decision starts. In some cases, one intermediate node in the route may not support one of the preferred metric types\(^1\). This may be because they have not installed the particular routing metric acquisition procedures prior to joining the network. In this case, the FRD framework can ignore this unsupported metric type in the routing decision. Although this case is not the desire situation, it should make a better decision than using a single routing metric type.

\(^1\) For example, node A, B and C are in the same route. Node A and B supports 3 metric types; Distance Vector, ETX and Trust metric, but node C supports only Distance Vector and Trust metric. The FRD framework will ignore ETX and only take Distance Vector and Trust metric into consideration as these two metric types are the common metric types of all nodes in the route.
Parts of the research work presented in this thesis have been published in the following conference proceedings.


1.5 Thesis Structure

Chapter 2 introduces routing procedures in MANETs. This chapter includes a discussion of the challenging issues in designing a routing protocol for MANETs. The chapter also includes the MANET routing protocols which are building blocks of our work in the later chapters. In addition, a survey of the security attacks in MANETs is also discussed in this chapter. The survey of the security attacks will be used to explain some security issues on the SETX routing protocol we present in Chapter 3.

Chapter 3 presents the SETX protocol to counter black hole attacks in MANETs. The SETX protocol is based on the ETX metric which is a common routing metric in MANETs. This chapter identifies security issues on the ETX protocol. It also describes the SETX protocol procedure and how it can be used to counter one of the security issues on the ETX protocol. A simulation study of this novel solution is also presented.

Chapter 4 presents a novel method, called the Flexible Routing Decision (FRD) framework. This framework is designed for cross-layer and multi-metric routing decision making in MANETs. It uses a cross-layer approach to support application-level QoS requirements by allowing users or software developers to use different routing metric types. The aim is to allow the network layer to make the most appropriate routing decision for a given application. This chapter also investigates how different routing metric types are used coherently to support diversified QoS requirements on a single platform. The
chapter also critically analyses the FRD framework and evaluates its performance using the NS2 simulation package.

Chapter 5 summarises the thesis. The chapter discusses the findings from the research. This includes a direction, issues found and methodology used during the research. It also includes the contributions and the discussions for future work.
Chapter 2

MANET Routing

2.1 Chapter Introduction

This chapter introduces MANET routing procedures. It describes the common processes of routing functions in MANETs. The chapter gives examples of state of the art MANET routing protocols and how they work. In addition, a security threat is another issue that cannot be ignored in MANET routings. Many of the existing MANET routing solutions were designed without taking security threats and attacks into consideration. They assume that all nodes are truthful. From this point of view, it allows an attacker to take advantage of the network through a routing solution. To be able to provide a secure solution, we need to understand the mechanism of these attacks. This chapter also discusses on threats and security attacks in MANETs routing.

Section 2.2 introduces the OSI Model. Section 2.3 discusses MANET routing and gives an example of a MANET routing protocol. Section 2.4 identifies security threats and attacks in MANET routing. Section 2.5 discusses the best way forward. Section 2.6 concludes the chapter.

2.2 OSI Model

Open Systems Interconnection (OSI) model is considered as the primary architectural model for computer networking [TAN11]. The OSI model describes how information or data transfers from application programmes (e.g. internet explorer) through a network medium (e.g. wired or wireless) to another application programme located on another network. The OSI model defines a division of network operations into 7 layers as shown in Figure 2.1.
The Application Layer

The application layer is the top layer of the OSI model which is closest to users. It allows applications to gain access to the network services. This application layer represents the services, that directly support the user applications e.g. file transfers, e-mail, and web browser.

The Presentation Layer

The presentation layer ensures that the data sent from the application layer of one computer (called a node) is readable by the application layer of another node. When data is sent out, the data is encoded into a generic format before the transmission. When data is received, the encoded data is decoded from the generic format to a format that is understandable to the application. The examples of the common communication services provided by the presentation layer are data encryption and text compression.

The Session Layer

The session layer defines how to establish, manage and terminate connections (called sessions) between applications from two nodes. The session layer also handles access control to allow only designated parties participating in the session.
**The Transport Layer**

The function of the transport layer is to accept data from the session layer. The transport layer splits the data up into smaller units (packet) and passes these to the network layer. This layer also ensures that the pieces all arrive correctly at the other end.

**The Network Layer**

The network layer controls the routing operation in the network. It defines how routing works and how routes are discovered. The network layer also decides how data packets are routed from the one node to another node.

**The Data Link Layer**

The data link layer provides error control and synchronization for the physical layer. In this layer, data packets are encoded and decoded into frames. It deals with transmission and handles errors in the physical layer including flow control and frame synchronization. The data link layer is divided into two sublayers: The Media Access Control (MAC) layer and the Logical Link Control (LLC) layer. The medium access control sublayer deals with the problem by providing channel access control mechanism. This makes it possible for several systems or nodes to communicate in a shared medium. The LLC layer controls frame synchronization, flow control and error checking.

**The Physical Layer**

The physical layer concerns the transmitting of raw bits over a communication channel. The design standards have to make sure that when one node sends a "1" bit, it is received by the other node as a "1" bit, not as a "0" bit. This raises questions on what electrical signals should be used to represent a "1" bit and a "0" bit, how many nanoseconds the bit lasts, whether transmission can be proceeded simultaneously in both directions, and how the initial connection is established. These design issues mainly deal with mechanical, electrical, and timing interfaces, as well as the physical transmission medium (e.g. radio frequencies in wireless network).
As mentioned in Chapter 1, the aim of this thesis is to design routing solutions that can satisfy application level requirements. That means our solution will require a mechanism to link between both the network layer and the application layer. The details of this cross layer routing solution will be discussed in Chapter 5. In the next section, we introduce MANET routing in order to understand how routing functions in the network layer works.

2.3 MANET Routing

2.3.1 MANET Routing Introduction

For better understanding, routing functions in MANETs can be explained in three main phases: route discovery, route maintenance and data forwarding shown in Figure 2.2. It starts with the route discovery phase where a node starts to find a route when it needs to communicate with another node. Once a route is found, the communication is started. Here begins the data forwarding phase. During the course of communication, if a link becomes unavailable, e.g. due to reasons such as node mobility (i.e. a node is moving out of the transmission range) or battery blackout, an alternative route should be sought for the communication to continue. The route maintenance phase deals with a broken route. In this stage, a node which detects a broken route will try to find an alternative route from the local cache. If there is no other route in the cache, the node which detects a broken route will initiate another route discover phase to find another route to the same destination node. This is the cycle of the process of routing in MANETs.

![Figure 2.2: MANET Routing Operation Cycle Model](image-url)
As mentioned above, the first routing stage, i.e. the route discovery stage, is the process by which one node finds a route to another node in the network. This process is initiated when a node joins a network or whenever a node (i.e. a source node) wishes to communicate with another node (i.e. a destination node). The source node asks its neighbours to find a route to the destination node. It typically does so by broadcasting a route request packet (i.e. RREQ) into the network. As shown in Figure 2.3, a source node S broadcasts the route request packet. The downstream nodes will rebroadcast this packet until the packet is received by the destination node.

![Figure 2.3: The Broadcasting of a Route Request Packet](image)

The destination node D will return a route reply packet (i.e. RREP), which concludes the route discovery stage (as shown in Figure 2.4).

![Figure 2.4: The Replying of an Acknowledgement Packet](image)
Upon the execution of the route discovery stage, more than one route may be found in the network (shown in Figure 2.5). If this is the case, the source node will select the best one from the available routes based on the routing algorithm used. For example, some routing protocols select the best route based on hop counts. In other words, these protocols select a route with the least hop count to the destination node. Others may select a route with the highest available bandwidth to the destination. For the example in Figure 2.5, there are two routes found. S will select Route 1 as it has lower hop count (i.e. 2 hops) than Route 2 (i.e. 3 hops).

Once the route has been found and selected, the selected route will be established. The data forwarding stage commences. This is the stage where nodes communicate with one another. However, if a mobile node detects a failure of an active link, the transmission will be suspended\(^1\). The upstream node will report this link breakage to mobile nodes en route from the source node to the destination one. Once the source node is notified, it can initiate another route discovery process. In other words, the more network topology changes, the more broken links there are. Then the network routing operation cycle will repeat more frequently.

---

\(^1\) All messages received may be buffered at intermediate nodes until a route to the same destination can be recovered.
2.3.2 Routing Metrics

As mentioned above, the best route is selected depending on the routing algorithm used, a selected routing metric type (which is selected by the routing algorithm) is the key to define the best route. Each routing metric type has its own aspect to measure a different quality of a link. One may measure the link delay, while another may measure how reliable (e.g. how much packet lost during the transmission) the link is. Some examples of routing metric types are as follows.

Distance Vector (Hop Count)

A distance vector metric type (or hop count) [PER94, PER97, JOH96] counts the number of hops from the source node to the destination node. For example, a route from a source node (S) to the destination node (D) is S→A→B→D. The hop count metric value of this route is 3. If this routing metric type is being used, the best route will be the one with the fewest hops. This metric type is one of the most common routing metric types in MANETs.

However, a route with the least hop count is not always the best route. If a route with a fewer hop count has a very low available bandwidth\(^1\), it may not perform as well as a route with more hop counts which has a higher available bandwidth. So a routing metric type which takes available bandwidth into account might perform better than the distance vector based metric type in this case.

Available Bandwidth Aware

An example of the available bandwidth aware metric type is [KIM10]. It is a routing metric designed to solve the traffic concentration area problem. The idea is to find available bandwidth which is computed as estimation with the total bandwidth minus the occupied bandwidth of each link on a node. If a link has a lot of available bandwidth, a node can transmit more data quantity through the link.

\(^1\) In this thesis, bandwidth refers to the maximum throughput that a link or a route can provide. On the other hand, available bandwidth means the throughput that can actually be used.
**Delay Aware**

An example of the delay aware metric type is Per-hop Round Trip Time (RTT) [ADY03]. The RTT metric measures the round trip delay between neighbouring nodes by using probes. To find the RTT, a node sends a probe packet carrying a timestamp to each of its neighbouring nodes on a fixed periodic basis. Once neighbouring nodes receive the probe, they respond with a probe acknowledgment which includes a timestamp. When the sending node receives the probe acknowledgement, it can find the estimate of the RTT.

The delay metric type does not just measure a delay but it also measures other aspects of link quality. For example, when either node or a neighbouring node is busy, the probe or the probe acknowledgement will experience queuing delay. As a result, the delay metric value will be higher.

**Expected Transmission Count (ETX)**

ETX [DEC03] estimates the number of transmission needed (including retransmissions) to successfully deliver a packet in a homogenous sensor networks. This is done by measuring the loss rate of broadcast packets\(^1\) (called probes) between the node itself and neighbouring nodes. ETX improves from Hop Count metric by taking packet loss rate into consideration. It may improve the performance from using a Hop Count metric, but however, does not consider load or a bandwidth level of a link. The use of the ETX metric will be discussed in Section 2.3.3.3

**Trust Metric**

The trust metric type is defined as a certain level of belief that a node regards another (neighbouring) node as reliable. A trust metric value of a node is usually estimated according to the information that is acquired directly, e.g. by observing the node’s past behaviour (called direct trust), or by recommendations from other neighbouring nodes (called indirect trust) [PIR04, PIR06, XUE04]. The higher the trust value a node assigns to a neighbouring node, the more reliable the assigned node is believed to be. This metric

---

\(^1\) Broadcast packet refers to a packet that is broadcast to neighbouring nodes within the coverage of the broadcast node.
type can also be used for finding routes with a higher level of reliability. An example of a routing protocol that utilises this routing metric type is Trust-based Routing Protocol [XUE04]. This protocol will be discussed further in Section 2.3.3.4.

**Remaining Battery Aware**

A node’s remaining battery capacity can reflect the life of a communication route. This metric type can be very important in mission critical applications such as military battlefield networks. An example of a remaining battery aware protocol is [SIN98]. The aim of this routing metric type is to maximise the life of mobile nodes the network.

**Battery Cost Aware**

This metric type estimates how much of the energy/battery consumption is required to transmit a packet. It avoids selecting a route with high energy consumption in order to preserve the energy of a node. This metric type can sometimes be used in conjunction with remaining battery aware to predict the lifetime of nodes according to the current traffic conditions [KIM02].

### 2.3.3 MANET Routing Protocols

#### 2.3.3.1 Destination Sequence Distance Vector (DSDV) Routing Protocol

The Destination Sequence Distance Vector (DSDV) protocol [PER94] is extended from the Internet infrastructure and table based Routing Information Protocol (RIP) [RFC1058]. RIP is based on the Distance Vector (DV) algorithm [MAL95]. It uses a hop count as a routing metric to select the best route to a destination. DSDV is recognised as the most famous routing protocol for MANETs. This is because its mechanisms (e.g. sequence number technique) have been applied to and become a standard in many routing protocols later on (e.g. AODV, DSR).
RIP is simple but effective and suited to small sized networks. However, it suffers from looping and count-to-infinity problems [CHE87]. In addition, as it was not designed to cope with dynamic network topological changes, RIP is not suited to MANETs. The DSDV routing protocol was designed to overcome these problems by adding a sequence number to each route in the routing table of RIP.

The value of the sequence number for a given route is incremented every time when the node (the origin node) broadcasts a routing information update. So the most recent route (the higher number\(^1\)) always has the highest sequence number, and a route with a more recent sequence number will always be chosen regardless of the route metrics. In other words, the sequence numbers are used to distinguish a new route from a stale route. If there exists multiple routes to the same destination and if these routes have the same sequence numbers, then the route with a better metric value (e.g. least hop count) is preferred [PER94, HE02].

DSDV also requires mobile nodes to periodically update their routing information stored in their routing tables. So routes are available when needed immediately thus reducing the latency in forwarding data packets.

### 2.3.3.2 Ad hoc On-Demand Distance Vector (AODV) Routing Protocol

AODV [PER97] is built on the Destination Sequence Distance Vector (DSDV) routing protocols [PER94, HE02]. It inherits the use of sequence numbers and the distant vector algorithm from DSDV. However, AODV differs from DSDV in that it does not maintain a complete list of routes in the network, rather it discovers routes only when they are needed.

With AODV, a route discovery process is initiated when two nodes in the network want to communicate to each other, but there is no known route between them. The process makes use of two types of control packets; Route REQuest (RREQ) packets and Route REPly (RREP) packets. The source node initiates the process by transmitting a RREQ packet that

---

\(^1\) Normally the most recent number is the higher sequence number. However, a sequence number can be reset if it reaches its limit. Therefore, the most recent number can be lower than a stale one.
will be relayed by intermediate nodes until it reaches the destination node that responds with a RREP packet. This process has been discussed earlier in Section 2.3.1.

AODV uses a route error (RERR) packet for route maintenance to cope with dynamic network topological changes. Once a mobile node detects a broken link, the node will propagate an unsolicited RERR packet to all the nodes already involved in the route discovery process. The RERR packet will be relayed until all active source nodes are notified. Upon receiving the notification of a broken link, the source node may restart another path discovery process if a path to the destination node is still needed.

In contrast to the DSDV routing protocol that advertises for route discoveries periodically, AODV initiates a route discovery process only when a route is needed. Therefore, the processing overhead introduced to mobile nodes and the unnecessary control traffic injected into the underlying network as the result of the periodical route advertisements are prevented when there is no topological change in the MANET [DIA06]. However, AODV has a larger initial latency compared to DSDV. This is because AODV does not obtain the route to the destination node when needed. It has to initiate a route discovery which introduces more delay before the communication can start.

### 2.3.3.3 Expected Transmission Count (ETX)

ETX [DEC03] metric is a routing metric that is used to find low loss rate routes in MANETs. The ETX of a route is the expected total number of packet transmissions (including retransmissions) required to successfully deliver a packet along that route. It is computed using a forward delivery ratio and a reverse delivery ratio of links along the route. The forward delivery ratio value is the probability that the data packet is successfully delivered at the neighbouring node. The reverse delivery ratio value is the probability that a node successfully receives a packet from the neighbour node. These probabilities are calculated by using a reply packet. Nodes exchange their probes to neighbours. It is required to wait for \( \text{START\_UP\_TIME} \) seconds, then nodes can calculate the delivery ratios to find the ETX metric of a link.

---

1 Nodes in the network have to wait for \( \text{START\_UP\_TIME} \) before the route can be established, otherwise,
Here, we will clarify the terms of *forward delivery ratio* and *reverse delivery ratio* first. In a link between two nodes, we pick one node as a focus node. Using the example from Figure 2.6, there is a link between node \( I \) and node \( J \) (i.e. \( I \rightarrow J \)). If we choose node \( I \) as our focus node, the delivery ratio of the link from the focus node \( I \) to node \( J \) \((d_{I \rightarrow J})\) is called as *forward delivery ratio*. On the other hand, the delivery ratio \((d_{J \rightarrow I})\) from the link from a neighbouring node of node \( I \) (i.e. node \( J \)) to node \( I \) will be *reverse delivery ratio*.

![Diagram](image)

**Figure 2.6: The difference between forward delivery ratio and reverse delivery ratio when focusing at different nodes.**

On the other hand, if we focus on node \( J \) instead of node \( I \). The *forward delivery ratio* is will be the delivery ratio from the link \( J \rightarrow I \) \((d_{J \rightarrow I})\). The *reverse delivery ratio* will be the delivery ratio of the link \( I \rightarrow J \) \((d_{I \rightarrow J})\). Since both delivery ratios are called differently when we focus at different nodes. From now on, we define a node that we focus as an initiator node.

*The procedure of the ETX protocol*

In order to make it easy to understand, we use Figure 2.7 to describe how initiator \( I \) and node \( J \) find the ETX metric value step by step. The procedure consists of 5 main steps: generating probe packets, advertising probe packets, calculating *reverse delivery ratio*, advertising *forward delivery ratio*, and calculating ETX metric value.

---

the Nodes do not acquire enough probe packets to judge whether the route has a high or low delivery ratio. This waiting time is implemented in the real tested base network from [DEC03].
Step 1, node $I$ generates a probe packet, $p_I$. Node $J$ also generates its probe packet, $p_J$. The probe packet simply contains information to indicate the origin of the packet. In this case, a probe packet contains the initiator node IP address.

Step 2, nodes broadcast their probe packets to their neighbouring nodes. This step begins when a node joins the network. The step will continue until the node leaves the network. The rate of the broadcasting probe packet ($BROADCAST\_RATE$) is 1 probe per second (as specified in the original ETX specification [DEC03]).

Step 3, after initiator $I$ received the first probe packets from node $J$ for $START\_UP\_TIME$ seconds, node $I$ will be able to calculate a reverse delivery ratio value (i.e. $d_{J \rightarrow I}$). The $d_{J \rightarrow I}$ is calculated from the $receiving\_rate$ divided by $BROADCAST\_RATE$ (as shown in Equation 2.1).

$$ reverse\_delivery\_ratio = \frac{receiving\_rate}{BROADCAST\_RATE} \quad \text{Equation 2.1} $$

The $receiving\_rate$ is the number of probe packets that nodes received ($n_{probe\_receive}$) within the last $START\_UP\_TIME$ seconds. The recommended $START\_UP\_TIME$ value is 15 seconds (as described in the original ETX test-bed [DEC03]).

If we set the value of $START\_UP\_TIME$ high, the ETX value will be more accurate. The reason is that nodes will need to receive more probe packets to have the same
receiving_rate. In other words, \( n_{probe_{receive}} \) needs to be higher when \( START_UP_TIME \) is higher, so that \( receiving_rate \) will be the same. If nodes need to wait longer, the \( receiving_rate \) will be more mature, thus more accurate.

At this step, node \( J \) will also calculate its reverse delivery ratio \( (d_{J \rightarrow I}) \). This reverse delivery ratio of node \( J \) is actually a forward delivery ratio of initiator \( I \). In other words, initiator \( I \) now has its reverse delivery ratio \( (d_{J \rightarrow I}) \), but node \( J \) has node \( I \)'s forward delivery ratio \( (d_{I \rightarrow J}) \).

Step 4, nodes exchange its reverse delivery ratio with neighbouring nodes. Once initiator \( I \) receives node \( J \)'s reverse delivery ratio \( (d_{J \rightarrow I}) \), it will store this value as its forward delivery ratio \( (d_{I \rightarrow J}) \). In a similar way, node \( J \) will store initiator \( I \)'s reverse delivery ratio \( (d_{J \rightarrow I}) \) as its forward delivery ratio.

Step 5, now node \( I \) and node \( J \) have both \( d_{I \rightarrow J} \) and \( d_{J \rightarrow I} \). The ETX metric of the link between \( I \) and \( J \) \((J \leftrightarrow J)\) can be calculated by,

\[
ETX_{I \rightarrow J} = \frac{1}{(d_{I \rightarrow J} \times d_{J \rightarrow I})} \tag{Equation 2.2}
\]

Please note that Equation 2.2 is used to find the ETX value of one link between node \( I \) and node \( J \) in a route in a homogeneous sensor network. As a route is constructed from several links (for example, route from source node \( S \) to destination node \( D \) may be constructed from \( S \rightarrow A \rightarrow B \rightarrow ... \rightarrow D \)), the ETX value of this route is the sum of the ETX metric values of all the comprised links. It accumulates the ETX metric values of each link from the source to the destination node. If the route \( S \rightarrow D \) consists of the links between \( I \) and \( J \), the ETX value of the route \( S \rightarrow D \) will be

\[
ETX_{S \rightarrow D} = \sum ETX_{I \rightarrow J} \tag{Equation 2.3}
\]
Routes with higher ETX value are likely to have more hops. The routes with more hops might have higher loss rate due to interference between hops of the same path [DEC03]. So, routes with a lower ETX metric value have a lower possibility that a packet will be lost. In a route selection process, a route with a lower ETX value is more preferable.

The ETX protocol considers a link asymmetry by calculating the reverse delivery ratio and forward delivery ratio in both directions. This is different from the AODV routing protocol. These typical routing protocols normally assume that the links are symmetric which is not always true in wireless networks. Many real world wireless links may be unidirectional. It may be because of the uneven transmission power or an obstacle between nodes. A link asymmetry is one of the unique challenges in wireless communication which many routing protocols do not address. However, the ETX protocol does address this challenge by considering the delivery ratio on both directions.

In an ideal case, two nodes may receive all probes between each other without missing a single probe. The delivery ratios of both upstream and downstream will be 1. The ETX value of this link is then calculated as 1 ($=\frac{1}{1x1}$). All of the data packets sent through the link are predicted to be delivered successfully. Thus a route with the lowest ETX value (the best in terms of both delivery ratios) is more preferable in a routing decision.

In reality, nodes may not receive all the probes. The delivery ratios can be under 1. This increases the ETX value. In the worst case, the link is broken and no probe packet is delivered. The ETX value will be considered as 10,000. It simply indicates that this link is not available.

The ETX approach also considers the hop count in a route. A route with a lower hop count has a lower loss rate than a route with a higher hop count due to two reasons; the interference between different links of the same route [BER87] and the risk of losing packets in a wireless communication.

However, routes with a higher number of hops with all perfect links may not always be more preferable than a route with one lossy link. By using the following situation as an example, a source node may choose a route to send its data as a one-way communication, when there are two routing candidates. The first is a three-hop route with perfect links (lossless). Its ETX value is 3 (from 1+1+1). The other route is a one-hop route with a 50%
delivery ratio. Its ETX value will be 2. The routing decision of the ETX approach will choose the second route. For the first route, the source node has to wait for the 2nd and 3rd intermediate nodes finishing their forwarding in order to avoid collision. On the other hand, the second route could provide a lower loss rate. The source node can simply retransmit when a packet is lost. Although the ETX approach takes the hop count into consideration, it still relies on the ETX sum of a route to select routes tending towards lower loss rate than alternative routes.

### 2.3.3.4 A Trust-based Routing Protocol (TRP)

A trust model was initially introduced in computer networks by [YAA94]. It was developed and modified for a MANET context by [BET94]. The original model makes it possible to take into account various types of trust relations in the routing functions in low mobility MANETs. The experiences that a node observes from another node are expressed in a trust value. The trust value can be calculated by monitoring, assessing and quantifying the truthfulness or the reliability of a neighbouring node with which it is associated. A Trust-based Routing Protocol (TRP) [XUE04] provides a routing algorithm based on trust models.

Here, a direct trust value is derived from positive and negative direct observation experiences between two neighbouring nodes. Each node passively observes the behaviours of its neighbours without using any interaction with other nodes. All direct trust values are initialised to 0 by default. However, nodes are free to initiate trust values to some other values if they have some pre-established trust relationships. For example, if node $i$ knows node $j$ prior joining the network. Node $i$ may initialise the trust value of node $j$ to 0.5. This helps node $j$ to become more trustable on node $i$ than other new incoming nodes.

The trust value of node $i$ on node $j$ is derived from the total number of positive and negative experiences that node $i$ observes on node $j$. A direct trust value ranges between 0 and 1. If node $j$ always behaves well, its direct trust value will eventually increase up to 1. If node $j$ is moderately malicious, its direct trust value is likely to be stable. If node $j$ is malicious, then its direct trust value will immediately become untrusted (0).
The trust models can be used to detect a malicious node which may not deliver traffic properly. They might drop a packet, modify a packet or impersonate other nodes. If a mobile node detects that their neighbours have any of the behaviours mentioned above, the node will reduce their trust values. The trust values of misbehaving nodes will be very low if they consistently misbehave. As a result, these misbehaving nodes will be ignored when routing decision are being made. In other words, nodes with very low trust values will be excluded in a chosen route.

2.4 Security Threats and Attacks

After we understand how routing in MANETs work, this section we discuss about security threats and attacks in MANETs. Since our aim is to design a routing solution efficiently, it is important to understand how security threats and attacks may affect the performance of a routing function.

Generally, security threats and attacks can be classified into passive and active attacks. In a passive attack, an attacker attempts to gain unauthorised access to confidential data by monitoring or capturing data transmitted over the network without disrupting the network operation. Examples of passive attacks in ad hoc networks include eavesdropping, traffic analysis and traffic monitoring. These types of attacks do not significantly affect the routing operation of the underlying network, but they can be a prerequisite for an active attack. Active attacks, on the other hand, could cause more disruptions to the underlying network operations. This is because, in an active attack, the attacker tries to alter, inject, delete or falsify authorised data, e.g. routing information. In addition, security attacks may be further classified into internal attacks and external attacks depending on the origin of an attacker. If an attacker is an insider of a communication group, domain or network, then the attacks launched by the attacker are usually called internal attacks. On the other hand, if the attacker is not a valid member of the communication group or network, the attacks are referred to as external attacks.

MANETs are vulnerable to many types of threats and attacks, which may occur at different layers in the OSI model. As this chapter focuses on security issues associated with routing in ad hoc networks, the discussions here are on the threats and attacks on routing protocols,
i.e. the network layer of the OSI model. The following section provides an analysis of how these attacks may be performed on ad hoc network routing protocols.

### 2.4.1 Eavesdropping Attacks

Eavesdrop or interception attacks are a form of passive attack. They occur when a malicious node intercepts messages sent between legitimate mobile nodes. The messages can either be routing information or user data. By intercepting user data, a malicious node may be able to learn some sensitive information about the users. However, if the malicious node intercepts routing information, the malicious node could analyse the underlying network infrastructure and traffic structure, launching further attacks and/or compromising data and location privacy.

IPSec [SSL3, X509] is an exemplar solution which can be used to protect the confidentiality of user data against this attack. Eavesdropping may not directly affect network operations, but it is often a prerequisite for more advanced attacks, e.g. black hole attacks which will be discussed further in Section 2.4.8.

### 2.4.2 Message Dropping Attacks

Message dropping attacks [XIE08] are a type of denial of service (DoS) attacks. In this attack, a malicious node discards messages which are expected to be forwarded to its downstream neighbours. These attacks may be performed with the intention of disrupting the underlying network operation or reserving resources (e.g. power or bandwidth) for its own use\(^1\). The malicious node may choose to discard all the incoming messages (in which case, the node is called a black hole node), or drop them selectively (called a grey hole node). Message dropping attacks are difficult to detect. This is because the attack resembles the same behaviour as that exhibited when a legitimate node switches between online and offline modes [XIE08].

\(^1\) This can be called a selfish node (Section 2.4.9).
Message dropping attacks affect the network availability. Since messages are dropped, network services may be disrupted. Another typical outcome of the attack is that it will reduce the performances of routing protocols and of the network. For example, the average network throughput and packet delivery ratio will be reduced as the result of the attacks.

2.4.3 Message Modification and Fabrication Attacks

In a message modification and fabrication attack, a malicious node modifies an incoming message or fabricates an invalid message, before injecting it into the network. If no integrity protection measure is taken, such attacks on routing (control) packets could lead to delays in establishing routes or even failures in routing operations, which, in turn, can result in network flooding, routing disruptions and/or DoS attacks.

2.4.4 Delay Attacks (Jellyfish Attacks)

Sometimes a mobile node with a malicious intention may not drop routing packets directly, but rather delay or disorder them before forwarding them on. These attacks will adversely affect routing efficiencies. They decrease end-to-end average throughputs and causing network congestions [AAD07]. The problems caused can be more severe for real-time or delay-sensitive applications. Another consequence of these attacks is that they can cause buffer overflows at mobile nodes [AAD04] thus leading to higher packet loss ratios.

2.4.5 Replay Attacks

In a replay attack [WIN05], a malicious node retransmits previously captured packets. Encrypting routing information packets may not be sufficient to thwart this attack, as an attacker does not need to know the contents of the packets in order to launch such attacks. Attackers may use these attacks to advertise an invalid stale route (i.e. by broadcasting a previously captured routing packet) causing routing delays or even disruptions.
2.4.6 Impersonation Attacks

Impersonation attacks are also called spoofing attacks. In such attack, a malicious node presents itself with an identity of another node in the network. This attack is one of the preparation steps in the process of conducting another attack. For example, the attacker may use a message modification attack to blackmail a victim node. As a result, other nodes may believe that the victim node is a malicious node and to exclude the node in routing operations.

2.4.7 Flooding Attacks

Most reactive routing protocols (e.g. AODV, DSR) are vulnerable to flooding attacks during a route discovery process [VEN09]. A malicious node may repeatedly inject mass bogus messages into the underlying network [YI05]. The bogus messages can be either false routing packets (e.g. false RREQs in AODV) or false data packets. The purpose of such attacks is to consume network or other nodes’ resources as much as possible, so as to disrupt normal network operations and to prevent legitimate nodes from communicating to each other. Although these bogus messages will be eventually dropped by their destination nodes, valuable resources would have been consumed by then.

Different from flooding attacks in conventional wired or infrastructural wireless networks, flooding attacks in mobile ad hoc networks [WAN07] can be further classified into three categories: RREQ flooding, user data flooding and authenticated message flooding.

1. **RREQ Flooding**: RREQ flooding attacks are performed during the route discovery process of ad hoc on-demand routing protocols (e.g. AODV and DSR routing protocols). Attackers broadcast RREQ packets containing invalid destination addresses. Because there are no destination nodes for these packets, they are forwarded across the network consuming network bandwidth and other nodes’ resources without any valid routes ever being discovered.

2. **User Data Flooding**: A malicious node may dispatch streams of (useless) user data packets to all nodes along a route to a victim destination node. This can result in the depletion of the available network bandwidth affecting communication capability among legitimate mobile nodes.
3. **Authenticated Message Flooding**: Non-secure routing protocols (e.g. DSDV and AODV) use a forwarding-first approach to routing [WAN07]. With this approach, intermediate nodes do not verify incoming messages before forwarding them to their down-stream neighbours, so these protocols are vulnerable to RREQ flooding and data flooding attacks. To overcome this weakness, secure routing protocols, e.g. ODSBR [AWE03], have been proposed. They employ a flooding suppression mechanism that requires each message to be signed by its originator, and verified by each of the immediate down-stream neighbouring nodes before being forwarded on along the route. This approach prevents bogus messages from flooding the network, as they are dropped by the first-hop intermediate node from the malicious node. However, according to the study in [GUR04], verifying a signature that is generated using the elliptic curve algorithm [JOH99] and with a 160-bit key typically takes up to a few seconds. If a malicious node sends a large number of signed messages into the underlying network, the nodes neighbouring with the malicious node will be busy with verifying these messages causing massive delays in forwarding legitimate messages. In this case, we say these nodes are suffering from an authenticated message flooding attack. In other words, for protocols using this authentication-first approach [WAN07], malicious nodes can still bombard some intermediate nodes with bogus authenticated messages though it is difficult for them to launch flooding attacks with RREQ packets or data packets. The outcome of authenticated message flooding attacks is that if the flooded intermediate nodes are too busy with verifying the attack messages, their buffers will soon overflow and cause packet loss.

### 2.4.8 Black Hole Attacks

The black hole attack [DEN02] is one of the well-known security threats in ad hoc networks. Routing protocols such as AODV [PER97] and DSR [JOH96] are vulnerable to this attack so for clarity, we now explain the mechanism of the black hole attack using the AODV protocol.

Black hole attacks have two properties. First, a black hole node exploits an ad hoc routing protocol (e.g. AODV) by advertising itself as having a valid route or the best route to a destination node, even though the black hole node does not have a route. The intention of
the black hole node is to intercept packets. Second, the black hole node may drop the intercepted packets in order to interrupt the network operations.

Typically, a black hole attack is performed with three steps; the eavesdropping, message modification and fabrication, and message dropping attack. Figure 2.8 shows the usual steps taken by a malicious node when it launches a black hole attack.

![Figure 2.8: The Black Hole Attack Model](image)

**Attack Step 1 - Eavesdropping Attack:** In order to launch a black hole attack, a malicious node needs to first establish itself as a relaying router for the traffic, i.e. it needs to take control of the traffic flow. As a route is established through a route discovery process, the malicious node will wait for a legitimate node to initiate a route discovery process. As RREQ packets are transmitted across the network, the malicious node can intercept and modify these RREQ packets regardless of their intended destinations, and/or fabricate RREP packets if necessary. The purpose of this attack at step 1 is to attract, as much as possible, future traffic to be forwarded via the malicious node.

**Attack Step 2 - Message Modification and Fabrication Attack:** As mentioned above, once RREQ packets are intercepted, the malicious node will modify them and/or fabricate the reply packets in order to establish itself as a key relaying node along a valid route, or to advertise itself as part of the shortest or a better route to a destination node [RAM03].
There are two possible ways of doing this: (1) by modifying RREQ packets (as shown in Figure 2.9), and (2) by fabricating RREP packets (as shown in Figure 2.10).

(1) **Divert traffic by RREQ Modification**

![Figure 2.9: RREQ modification](image)

Legend
- Source node
- Intermediate node n
- Destination node
- Malicious node
- A link between two nodes
- Legitimate RREQ packets from the source node
- Modified RREQ packets sent by the malicious node

Figure 2.9 shows that a malicious node, M, receives a RREQ packet, and modifies it, possibly with false information, before forwarding it on. A commonly seen modification attack on a distance vector algorithm (e.g. on the AODV protocol) is to modify the hop-count value. In this case, M replaces the real hop-count value contained in the RREQ packet (which is 2) with a zero hop-count value. So when Node 3 compares the zero hop-count value contained in this modified RREQ packet (received from M) with the value in the packet forwarded from Node 2, Node 3 will discard the route via Node 2, and take on the route via Node M. This is because, according to the AODV specification, among the RREQ packets with the same sequence number, the one containing the lowest hop-count will be selected and forwarded to the destination, and the rest will be discarded. In other words, as the result of this modification attack by M, the legitimate RREQ packet is discarded by node 3 and the falsified RREQ packet is received by the destination node, D, and finally a route via the malicious node is established.
(2) **Divert traffic by RREP Fabrication**

![Diagram](image)

**Legend**
- Source node
- Intermediate node \( n \)
- Destination node
- Malicious node
- A link between two nodes
- Legitimate RREQ packets from the source node
- Fabricated RREP packets sent from the malicious node

**Figure 2.10: RREP modification**

As shown in Figure 2.10, M does not have a valid route to the destination node, D. However, upon the receipt of the RREQ packet, M fabricates a RREP packet and unicasts it back to the source node, S. Like the case in modifying the RREQ packet, this fabricated RREP packet contains a hop-count value of zero, which can make S to believe that the route through M is the shortest valid route to D.

**Attack Step 3 - Message Dropping Attack:** Once the route via the malicious node is established, the malicious node can launch a message dropping attack by refusing to forward all the subsequent data traffic sent along the route thus disrupting the communication between S and D. Optionally, the malicious node may choose to selectively discard the data packets, launching the so called "grey hole" attack [AGR08].

The grey hole attack is more subtle than the black hole attack, as in the former case, the malicious node may sometimes behave as a decent or legitimate node, and this makes the detection of the grey hole node much more difficult. Existing solutions attempting to thwart countering these threats include the work in [DEN02] that aims to detect and address a black hole attack committed by a single node, and [AGR08] that tries to tackle the attack committed by more than one black hole nodes that collaborate together to launch the attack. Malicious nodes may also analyse the conversation taking place between S and D if the underlying traffic is not encrypted.
2.4.9 **Selfish Nodes**

There is another type of risk imposed on the normal operations of MANETs, which is caused by the selfishness of some of the mobile nodes. As a MANET does not have any infrastructural support or dedicated gateways or routers to support traffic relay and delivery in the network, it has to rely on the goodwill and routing support by all the nodes in the network for effective and efficient network operations. However, there may be nodes that are selfish and refuse to relay traffic in order to reserve energy and resources for their own use. Such nodes are called selfish nodes (or passive selfish nodes) in MANETs. Though selfish nodes [MAR00] are different from other types of malicious nodes (e.g. black hole nodes) in that their refusal to relay traffic is due to selfishness, rather than any malicious intent, the outcome of such refusals are the same, i.e. the network cannot operate efficiently and effectively. The passive selfish nodes can further be classified into two categorises;

(1) **Passive selfish nodes that refuse to forward (drop) routing packets.** These selfish nodes hide their existence from their neighbours by dropping all routing packets they receive. The aim of these selfish nodes is to avoid being included in a communicating path to support communications among other nodes [YOK06]. They may force other nodes to communicate via a route with an unnecessarily high hop count value increasing packet delays and decreasing network throughputs.

(2) **Passive selfish nodes that refuse to forward data packets.** The selfish nodes in this case only forward routing packets truthfully, but drop data packets. Comparing with the selfish nodes that drop routing packets, the selfish nodes in this case are easier to detect as they show up their existence when forwarding the routing information packets.

For active selfish nodes, they actively make the route through themselves unattractive. They may use similar techniques as in black hole attack but instead of fabricating a better routing metric value, the active selfish nodes fabricate a routing metric with a worse value. So a route which has a better metric value would be selected.
Routing in MANETs has been an active research area. Although numerous routing protocols have been introduced for MANETs, there is no all-in-one solution for MANET routings. There are many application requirements in such networks. Some applications might require a high throughput (e.g. Videoconferencing). Some might need high security (e.g. Military communication). These requirements (high performance and high security level) have trade-offs between them. For example, a protocol may be designed with a very strong encryption algorithm to protect routing information. Although the routing information may be difficult to break and be eavesdropped, such a strong algorithm may require high computing power to encrypt and decrypt the data. Since nodes in MANETs might not be powerful, it could introduce more delay to the routing operation. On the other hand, if the protocol has no encryption algorithm or a very weak one, the routing operation may be faster, but it may not be able to protect the data against security attacks. It is difficult to find a balance between two or more different requirements of different applications. Therefore, this makes it harder to design a MANET routing protocol.

In addition, MANET routings face many challenges, such as limited resources (e.g. processing power, bandwidth, and storage), node mobility and limited physical security. The major issues that affect the design, deployment and performance of MANETs may include medium access scheme, routing, multicasting, quality of service provisioning, self-organisation, security, energy management, scalability, deployment consideration [MIC02]. One of the most important issues in MANET routing is security. The absence of a central manager, limited resources and shared wireless medium makes MANETs more vulnerable to an attack than a traditional wired network. In the traditional wired network, routers within the central parts of the network are owned by a few well known entities and are therefore assumed to be trustworthy. This assumption no longer holds in MANETs since all nodes entering the network are expected to take part in routing. Also links between nodes are connected using wireless as a medium. Instead of physically tapping the line, the communication in MANETs can be eavesdropped just by being in promiscuous mode (i.e. listening). In addition, the topology in such a network can be highly dynamic. Traditional routing protocols can no longer be efficient in this case. These entire reasons can make designing a routing protocol in MANETs much more difficult than the traditional wired networks.
2.6 Chapter Summary

This chapter introduced the routing in MANETs. It explained common processes of routing in MANETs, and the different approaches which could be used in the MANET routing protocol. Several popular MANET routing protocols were also discussed and explained how they work. It also gave a discussion about the difficulty in designing a routing protocol in MANETs. A survey of the threats and attacks on MANETs routing was also discussed in this chapter. The attacks include eavesdropping attacks, message dropping attacks, message modification and fabrication attacks, delay attacks, replay attacks, impersonation attacks, flooding attacks, black hole attacks and selfish nodes. Each attack requires different security mechanisms to prevent or detect. The next chapter, we will discuss on a security solution purposed to secure the ETX protocol.
Chapter 3

A Countermeasure to Black Hole Attacks in Mobile Ad hoc Networks

3.1 Chapter Introduction

This chapter presents a solution to counter black hole attacks in mobile ad hoc networks (MANETs). Black hole attacks are effective DoS (Denial of Service) attacks committed by fabricating routing information, attracting packets to route through the attackers. Once a black hole node has gained the control over the traffic, it can monitor, alter or drop the traffic. The ETX (Expected Transmission Count) measures the delivery ratio of a wireless link so that a routing solution can use the metric to find routes with a low packet loss rate. However, the acquisition of an ETX metric value is open to abuse. This chapter describes a solution to counter the black hole attacks on the ETX metric acquisition process (or during the route discovery phase). The solution is called the Secure ETX (SETX) protocol. Instead of allowing individual nodes to advertise their respective delivery ratios and acquire the metric values from other nodes (as in ETX) at will, the protocol allows nodes to measure neighbours’ delivery ratios directly. Simulation results show that this novel protocol can provide a marked improvement in network performances in the presence of black hole attacks that fabricate routing information, and it can do so with a negligible level of additional overhead.

This chapter is structured as follows. Section 3.2 analyses security attacks on the original ETX protocol. Section 3.3 discusses our design of the SETX protocol. Section 3.4 analyses security attacks on the SETX protocol. Section 3.5 reviews existing solutions to counter black hole attacks and compares the SETX protocol with the existing solution that was designed to counter black hole attacks on the ETX protocol. Section 3.6 evaluates the protocol using simulation studies. Section 3.7 discusses limitations of the SETX protocol, and finally Section 3.8 concludes the chapter.
3.2 ETX Security Analysis

This section discusses how ETX can be vulnerable to security attacks, including how security attacks may be performed and the implications of the attacks. The discussion focuses on the attacks on the original ETX protocol specification [DEC03]. There are three attack cases: Case 1: advertising a falsified forward delivery ratio; Case 2: modifying BROADCAST_RATE, and Case 3: impersonation attacks.

3.2.1 Case 1: Advertising a Falsified forward delivery ratio

In this attack, a malicious node, M, falsifies a forward delivery ratio in an attempt to modify the ETX metric value. Depending on the purpose of the attack, the malicious node may modify the actual value of the forward delivery ratio into a higher or a lower value. The procedure and the purposes of the attack are discussed below.

![Figure 3.1: Advertising a falsified forward delivery ratio - Step 1, nodes advertise their probe packets](image)

Step 1: Initiator I broadcasts its probe packets (pI) to its neighbouring nodes, including malicious node M and neighbouring node A (shown in Figure 3.1). In the meantime, nodes M and A also advertise their probe packets to initiator I as well (i.e. pM and pA). Initiator I calculates reverse delivery ratios for the link from M to I (dM→I) and the link from A to I (dA→I), respectively. The reverse delivery ratios are calculated from the receiving_rate / BROADCAST_RATE (see Equation 2.1).

1 To avoid repetition, we will not discuss the procedure of the ETX protocol in this section (see Section 2.3.3.3 for the details of ETX protocol).
Step 2: Initiator $I$ requests the *forward delivery ratio* values for the link from initiator $I$ to node $A$ (i.e. $d_{I \rightarrow A}$), and the link from initiator $I$ to node $M$ (i.e. $d_{I \rightarrow M}$) from the corresponding neighbouring nodes\(^1\) as shown in Figure 3.2. Node $A$ sends a truthful $d_{I \rightarrow A}$ value. However, node $M$ may not be honest (e.g. it may want to be selected en-route). So it advertises a falsified $d_{I \rightarrow M}$ (called $d_{I \rightarrow M}^*$).

\[\text{Figure 3.2: Advertising a falsified forward delivery ratio} - \text{Step 2, node M advertises a falsified forward delivery ratio (}d_{I \rightarrow M}\text{) to I}\]

Step 3: Initiator $I$ now has $d_{I \rightarrow A}$ and $d_{I \rightarrow M}$ values. Figure 3.3 shows that initiator $I$ calculates the ETX values for these links based on $d_{I \rightarrow A}$ and $d_{I \rightarrow M}$ values. The ETX metric value calculated for the link between initiator $I$ and node $A$ (i.e. $I \leftrightarrow A$) is a truthful value as node $A$ is an honest node. However, the ETX metric value of the link between initiator $I$ and node $M$ (i.e. $I \leftrightarrow M$) is not truthful. This is because node $M$ fabricates the $d_{I \rightarrow M}^*$ value.

\[\text{Figure 3.3: Advertising a falsified forward delivery ratio} - \text{Step 3, node I calculates the ETX values}\]

\(^1\) Taking node $I$ as the initiator node, *reverse delivery ratio* $d_{A \rightarrow I}$ of node $I$ is calculated by node $I$ itself at Step 1, but *forward delivery ratio* $d_{I \rightarrow A}$ of node $I$ is calculated by the neighbouring node $A$ at Step 2.
Here there are two sub-cases. In sub-case 1 (i.e. Case 1.1), node $M$ advertises a better $d_{I\rightarrow M}^*$ value than the actual value, and in sub-case 2 (i.e. Case 1.2), node $M$ advertises a worse $d_{I\rightarrow M}^*$ value.

**Case 1.1:** Node $M$ advertises a $d_{I\rightarrow M}$ value that is better than the actual value.

Node $M$ wants to be included en-route. It falsifies routing information by modifying a routing metric value in order to be included en-route. In this case, Node $M$ advertises a $d_{I\rightarrow M}^*$ value that is better than the actual value. As the ETX value is calculated from $d_{I\rightarrow M}$ and $d_{M\rightarrow I}$, the resulting ETX value will be better\(^1\) than the actual value. A better ETX value will give node $M$ a better chance to be selected en-route. If the malicious node is included en-route, it can take control over the traffic and can launch attacks (e.g. by dropping data packets to interrupt the communication as discussed in Section 2.4.2).

Assuming that `BROADCAST_RATE` is 1 probe packet per second. During the last 10 seconds, initiator $I$ has broadcast 10 probe packets. Say Node $A$ has received 8 out of the 10 probe packets from initiator $I$, and node $M$ has received only 5 probe packets from initiator $I$. So the receiving rates of node $A$ and node $M$ are 0.8 and 0.5 probe packets per second, respectively. Then the forward delivery ratio from initiator $I$ to node $A$ (i.e. $d_{I\rightarrow A}$) will be $\text{receiving rate} / \text{BROADCAST RATE} = 0.8/1 = 0.8$, and similarly $d_{I\rightarrow M}$ will be $0.5/1 = 0.5$. In the meantime, initiator $I$ receives all the probe packets sent by node $A$ and node $M$, so both $d_{A\rightarrow I}$ and $d_{M\rightarrow I}$ will be 1 (i.e. $\text{receiving rate} = \text{BROADCAST RATE}$).

When initiator $I$ requests the forward delivery ratios from node $A$ and node $M$, node $A$ will send the actual value of $d_{I\rightarrow A}$ (i.e. 0.8) to initiator $I$, but node $M$ will not send the actual value of $d_{I\rightarrow M}$ (i.e. 0.5). Rather, it sends a fabricated value, say 1.0, to initiator $I$. As the ETX values of the link $I\leftrightarrow A$ and $I\leftrightarrow M$ are calculated by $1 / (d_{I\rightarrow A} \times d_{A\rightarrow I})$ and $1 / (d_{I\rightarrow M} \times d_{M\rightarrow I})$ (from Equation 2.2), respectively, the ETX values of the link $I\leftrightarrow A$ and $I\leftrightarrow M$ will be $1 / (0.8 \times 1) = 1.25$ and $1 / (1 \times 1) = 1$. As a lower value of ETX is preferred, initiator $I$ will select the malicious node $M$ en-route because the ETX value of the link $I\leftrightarrow M$ is better than

---

\(^1\) The lower the ETX metric value means the better the link capacity as measured by the ETX metric (so, in this thesis, we say the better the ETX metric value). A lower ETX value indicates that either or both delivery ratios are higher.
the ETX value of the link $I \leftrightarrow A$. This will give node $M$ the opportunity to monitor, drop or modify the traffic that passes through it.

**Case 1.2:** Node $M$ advertises a $d_{I \rightarrow M}$ value that is worse than the actual value.

This is opposite from the previous case. In this case, Node $M$ is selfish\(^1\), it wants to be excluded from the route selection to preserve its resources (e.g. battery power, computational power). So node M falsifies the *forward delivery ratio* (i.e. $d_{I \rightarrow M}$) with a lower value. Since the *forward delivery ratio* is lower, the ETX metric value becomes worse. As a result, initiator $I$ may not select node $M$ in the routing decision.

In some cases, a selfish node may be part of the route with a lower or the lowest loss rate. If it fabricates a lower *forward delivery ratio*, the route may not be chosen (i.e. a route with a higher loss rate may be chosen). In the worst case, there may be only 1 route available to a destination node. If the selfish node is located en route, and if it is selfish, say switches itself off or reports that the link is unusable (by advertising a zero $d_{I \rightarrow M}$ value), then there will not be any route to the destination, causing network partitions.

These attacks (i.e. Case 1.1 and Case 1.2) are possible because the ETX protocol allows a neighbouring node to calculate a *forward delivery ratio* and to advertise it to the initiator node without any measure to ensure that the advertised value is truthful. To thwart these attacks, the Secure ETX (SETX) routing protocol is proposed. The protocol shifts the task of calculating a link’s *forward delivery ratio* from the receiving (i.e. neighbouring) node to the initiator. It requires the neighbouring node to return probe messages contained in the probe packets they received. The SETX protocol also introduces a built-in mechanism to ensure that it is hard for the neighbouring node to forge the probe messages. Based on the returned probe messages versus those sent out, the initiator node calculates both *reverse delivery ratio* and *forward delivery ratio*. The detailed of the protocol will be discussed in Section 3.3.

\(^1\) This behaviour (i.e. avoid be included in a routing activity) is called selfish attack (See Section 2.4.9).
3.2.2 Case 2: Modifying \textit{BROADCAST\_RATE}

Another way to falsify an ETX value is by modifying the \textit{BROADCAST\_RATE}. When a malicious node modifies this value, the initiator will receive a varied number of probe packets (more or fewer than it is supposed to receive). The \textit{receiving\_rate} (i.e. the rate at which initiator \(I\) receives probe packets from node \(M\)) will also change. As \(d_{M\rightarrow I}\) is calculated using \textit{receiving\_rate} divided by \textit{BROADCAST\_RATE}, \(d_{M\rightarrow I}\) will be falsified as well. As a result, the ETX metric value will be falsified (higher or lower than the actual value). The procedure of this attack is further explained below.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{fig3.4}
\caption{Modifying \textit{BROADCAST\_RATE} – Step 1, initiator \(I\) and node \(A\) broadcast probe packets at an agreed rate, \textit{BROADCAST\_RATE}.}
\end{figure}

\textit{Step 1}: Initiator \(I\) broadcasts its probes (\(p_I\)) to its neighbouring nodes, including node \(M\) and node \(A\) (as shown in Figure 3.4). Meanwhile, node \(A\) also advertises its probes (i.e. \(p_A\)) to initiator \(I\). All the (honest) nodes broadcast their probes at an agreed rate of \textit{BROADCAST\_RATE}, which is a legitimate behaviour.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{fig3.5}
\caption{Modifying \textit{BROADCAST\_RATE} – Step 2, node \(M\) modifies \textit{BROADCAST\_RATE} and broadcasts probe packets at an illegitimate rate.}
\end{figure}
Step 2: However, node $M$ does not broadcast probe packets at the agreed rate of $BROADCAST\_RATE$ shown as a dotted line in Figure 3.5. It may modify this rate to serve its purpose - selfish attacks or black hole attacks. The details of these attacks on $BROADCAST\_RATE$ are explained below.

![Diagram](image)

**Figure 3.6: Modifying $BROADCAST\_RATE$ – Step 3, Initiator $I$ derives an incorrect reverse delivery ratio.**

Step 3: Initiator $I$ calculates an incorrect reverse delivery ratio ($d_{M\rightarrow I}^*$) as shown in Figure 3.6.

Here node $M$ modifies its $BROADCAST\_RATE$. This will lead to the modification of the delivery ratios of the links in the direction from node $M$ to its neighbouring nodes. In our example, this will reduce the values of $d_{M\rightarrow I}^*$ and $d_{M\rightarrow A}^*$ (Case 2.1) or increase the values (Case 2.2). It is worth noting that the delivery ratios of the links in the other direction (i.e. $d_{I\rightarrow M}$ and $d_{A\rightarrow M}$) will still be truthful.

**Case 2.1: Reduce $BROADCAST\_RATE$ (Selfish attacks)**

Node $M$ reduces the $BROADCAST\_RATE$ value, and advertises its probe packets at this reduced rate. So initiator $I$ will receive fewer probe packets. Initiator $I$ may believe that this is due to poor channel conditions. The reverse delivery ratio ($d_{M\rightarrow I}^*$) calculated by $I$ will be lower than the truthful value. As a result, the ETX value of the link will be worse than the actual value.

$BROADCAST\_RATE$ can be as low as 0. A zero value means that node $I$ has not received any probe packets sent by $M$. This could mean (a) $M$ has broadcast some probe packets, but none of them has reached to $I$ (this case is less likely if the number of probe packets broadcast is sufficiently large), or (b) $M$ has not broadcast any probe packets, e.g. it may
be running out of battery power or have switched itself off to preserve its resources. The last case is called passive selfish attacks (see Section 2.4.9). No nodes can communicate with them until they switch themselves back on. The wireless link through this node is not usable. The initiator $I$ will need to find another route if it still wants to establish a communication channel.

In addition to passive selfish nodes, an active selfish node does not switch themselves off, but rather it broadcasts probe packets at a much lower rate, resulting in a poor ETX value for the link. If the metric value is worse than the values of other routing candidates, the route containing the selfish node will not be selected. The initiator may select a sub optimal route.

A countermeasure to the selfish attacks is to use a trust model [XIE08] to encourage all the nodes in the network to actively participate in the routing operations.

**Case 2.2: Increase $BROADCAST\_RATE$ (Black hole / Flooding attacks)**

Node $M$ wants to be included en-route. It increases $BROADCAST\_RATE$ to $BROADCAST\_RATE'$ at his end, i.e. $M$ broadcasts probe packets at a rate higher than the rate that has been agreed with $I$. In this case, the number of probe packets received by $I$ (i.e. the receiving rate measured by initiator $I$) will go higher. The reverse delivery ratio ($d_{M\rightarrow I}$) and the ETX value will also be better. As long as the resulting reverse delivery ratio ($d_{M\rightarrow I}^*$) is less than or equal to 1 (e.g. when the link condition is lousy, and when there are a sufficient number of probe packets lost during transmission) and as long as the resulting ETX value for the link is better than the values measured for other candidates. Initiator $I$ will select node $M$ en-route.

However, if the link condition is good, the use of $BROADCAST\_RATE'$ may lead to receiving rate being larger than $BROADCAST\_RATE$, and the reverse delivery ratio measured by initiator $I$ being larger than 1 (as reverse delivery ratio = receiving rate / $BROADCAST\_RATE$). In this case, the reverse delivery ratio will make the ETX value better (as ETX value = 1 / (reverse delivery ratio × forwarding delivery ratio)). Initiator $I$ should notice that something is wrong with the reverse delivery ratio value. It should avoid selecting this route. However, according to the original implementation of the ETX protocol [DEC03], there is no evidence that there is a mechanism to detect whether the
reverse delivery ratio is higher than 1 or not. So if the initiator \( I \) follows the protocol specification strictly, this route should be selected as it should have a better ETX value than the other candidates.

It is worth noting that, in addition to (illegitimately) increasing the \( BROADCAST\_RATE \) by \( M \) (i.e. Case 2.2), there is another case where the \( \text{reverse delivery ratio} \) measured by initiator \( I \) may go higher than 1, i.e. when node \( M \) is a victim of an impersonation attack. In this case, another malicious node impersonates node \( M \) by broadcasting probe packets using \( M \)'s identity. Without any protective measure, initiator \( I \) will not be able to differentiate the probe packets sent by the impersonator from those from \( M \). As a result, the \( \text{reverse delivery ratio} \) will be higher than 1. This case, denoted as Case 3, is further discussed in the next section.

### 3.2.3 Case 3: Impersonation Attacks (Active Selfish Attacks)

The purpose of this attack is the same as that described in Case 2.1 (i.e. the selfish attack). In this attack, a malicious node advertises probe packets using another node’s identity (e.g. using initiator \( I \)'s identity) in an attempt to increase the ETX value of a link via another node so as to prevent itself from being chosen by the initiator. The following describes the process of this attack.

![Figure 3.7: Impersonation Attacks - Step 1, nodes advertise their probe packets](image)

**Step 1**: Initiator \( I \), node \( A \) and node \( M \) advertise their respective probe packets (shown in Figure 3.7).
Step 2: Node $M$ impersonates initiator $I$ by advertising probe packets using $I$’s identity ($p_I$) to node $A$ (shown in Figure 3.8). Node $A$ will receive two sets of probe probes. One set are genuine probe packets that are sent by initiator $I$, and another set are falsified probe packets that are sent by $M$. However, as the ETX protocol does not provide origin authentication, so $A$ would regard both sets of probe packets as from initiator $I$ when $A$ calculates the forward delivery ratio ($d_{I\rightarrow A}$). As a result, $A$ will derive an inflated forward delivery ratio for the link from $I$ to $A$ (i.e. $d'_{I\rightarrow A}$).

Step 3: when initiator $I$ request a forward delivery ratio from node $A$, node $A$ sends the incorrect (i.e. inflated) forward delivery ratio ($d'_{I\rightarrow A}$) to initiator $I$ (shown in Figure 3.9).
Step 4: initiator $I$ calculates an ETX value (shown in Figure 3.10). Since $d_{I \to A}$ is not truthful, the calculated ETX metric value will not be truthful too.

As discussed in Case 2.2 above, if the link between $I$ and $A$ is sufficiently lousy to bring the link’s reverse delivery ratio down to less than or equal to 1, and if the resulting ETX value of the link between $I$ and $A$ is better than the ETX value for the link via $M$, node $A$ will be chosen. In this way, node $M$ can avoid performing the routing operations.

However, if the link is in a good condition, the receiving_rate measured at $A$ will be higher than BROADCAST_RATE. In this case, the $d_{I \to A}$ value will be larger than 1. Node $A$ will realise that something is wrong in this ETX calculation. $A$ may believe that initiator $I$ is malicious, as $A$ may believe that $I$ has increased the BROADCAST_RATE on its end. However, in this case, initiator $I$ did not increase the BROADCAST_RATE; it was node $M$ who impersonated initiator $I$ and increased the number of probe packets received by $A$. In other words, in this case, node $A$ can detect that there are some fraudulent activities in the network, but $A$ cannot pin down who is the perpetrator.

### 3.2.4 ETX Security Analysis Summary

It can be seen, from the above discussions, that the original ETX protocol is vulnerable to a number of security attacks. If we categorise these attacks by their purposes, we can group them into two main categories: black hole attacks, and selfish attacks as shown in Figure 3.11.
For the black hole attacks, a malicious node wants to be included en-route to control the traffic. It may advertise a falsified forward delivery ratio (Case 1.1) or increase the BROADCAST_RATE (Case 2.2). These malicious actions will lead to a falsified ETX value that is better than the truthful value. This better ETX value can give the malicious node a better chance to be selected en-route. Once being selected, it can launch further attacks (e.g. packet dropping attacks).

Selfish attacks, on the other hand, lead to an opposite outcome from black hole attacks. Selfish nodes avoid being selected by advertising a lower forward delivery ratio (Case 1.2), decreasing BROADCAST_RATE (Case 2.1), and/or impersonating the identity of another node when sending the probe packets (Case 3). These illegitimate actions will reduce the ETX metric value of the route containing the malicious node. If this value is lower than the ETX value of another route, then the other route will be chosen (though, in fact, the route containing the malicious node performs better).

From the above analysis, the original ETX protocol is vulnerable to a number of security attacks. This is largely due to the fact that the protocol requires a neighbouring node to calculate and advertise a forward delivery ratio of the link between an initiator and the neighbouring node. There is no mechanism for the initiator node to verify whether or not a received forward delivery ratio is truthful. This gives a black hole node or a selfish node an opportunity to fabricate and advertise false forward delivery ratios to its neighbours.

The next section describes a new countermeasure to the black hole attack, i.e. the Secure ETX (SETX) protocol. The idea used in the design of the SETX protocol is that, rather than letting a neighbouring node to generate and advertise a forward delivery ratio for the link, the protocol requires the neighbouring node to return the probe packets received from an initiator node back to the initiator, and the initiator to calculate the forward delivery ratio based on the returned probe messages versus those sent out. In this way, it is harder
for a neighbouring node to forge a *forward delivery ratio*, as, for doing so, the node would have to provide the necessary probe messages.

It is worth noting that the proposed SETX protocol is not designed to address the issue of selfish attacks. An effective countermeasure to selfish attacks is to use a trust model (see Section 2.3.3.4 for more details) to encourage selfish nodes to actively take part in routing operations.

### 3.3 The Secure ETX (SETX) Protocol

#### 3.3.1 Protocol Overview

The SETX protocol is designed to counter security attacks described in Case 1 in Section 3.2.1. As discussed in Section 3.2.1, the *forward delivery ratio* acquisition of the ETX protocol is subject to abuse, so the design of the SETX protocol was focused on securing the acquisition of *forward delivery ratios*. Instead of allowing a neighbouring node to calculate and advertise the *forward delivery ratio* of the link connecting an initiator and the neighbouring node, the protocol introduces a mechanism for the initiator to measure the delivery ratio itself.

<table>
<thead>
<tr>
<th>Protocols</th>
<th>Reverse delivery ratio</th>
<th>Forward delivery ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETX</td>
<td>An initiator node calculates itself.</td>
<td>Neighbouring nodes calculate and advertise the value to the node.</td>
</tr>
<tr>
<td>SETX</td>
<td>An initiator node calculates itself.</td>
<td>An initiator node calculates itself.</td>
</tr>
</tbody>
</table>

*Table 3.1: The comparison of ETX and SETX protocols*

There are two main differences between the original ETX protocol [DEC03] and the SETX protocol presented here. The first difference is that, as described above, in the original ETX method, a *forward delivery ratio* is calculated and advertised by a neighbouring node, but in the SETX protocol, the *forward delivery ratio* will be calculated by an initiator node itself. This means there will be no *forward delivery ratio* sent from neighbouring nodes anymore.
The second difference lies in how probe packets are processed and used. In ETX, an initiator generates probe messages and broadcasts them using probe packets\(^1\). The node does not store the probe messages once they are broadcast. The neighbouring node of the initiator only records the number of probe messages or packets received; it does not store the probe message and retransmit them. However, in SETX, the initiator generates probe messages and broadcasts them using probe packets, and it also records all the probe messages that have been broadcast. It will use these probe messages to verify if the neighbouring node has really received them, and if so, how many. These statistics will be used by the initiator to calculate the forward delivery ratio for the link.

Here is an overview of the SETX protocol. The initiator \(I\) broadcasts probe messages to its neighbouring node \(J\). Node \(J\) then sends the probe messages back to initiator \(I\) to confirm that it has really received the probe messages from initiator \(I\). Initiator \(I\) then checks the number of probe messages that are received from node \(J\) and are identical to those sent by the node itself. Based on the number of positively verified probe messages, node \(J\) calculates a forward delivery ratio. This calculation disregards those ‘bad’ probe messages that are too old, missing or do not match with those sent. The more the ‘bad’ probe messages, the lower the resulting forward delivery ratio and the worse the ETX value. With this technique, to inflate (i.e. to maliciously increase) the forward delivery ratio for the link, the neighbouring node would have to forge and return a series of probe messages that could pass the check performed by the initiator. However, to employ this technique, each node is required to record/store every advertised and received probe messages.

To store both advertised and received probe messages, two types of buffers are used: an Advertised Probe Buffer (APB) and a Received Probe Buffer (RPB). APB is a probe buffer used to store probe messages that have been advertised, and RPB is used to store the probe messages received from a given neighbouring node.

\[^1\] A probe message is a content contained in a probe packet. It will be discussed further in Section 3.3.
Typically, a node will have one APB, used to store the probe messages advertised by the node itself, but several RPBs, one for each neighbouring node. In other word, a RPB is used to store the probe messages received from a particular neighbouring node. The number of RPBs maintained at a node is determined by the number of neighbouring nodes that the node connects to. For example, as shown in Figure 3.12, node I has two neighbours: nodes A and B, so node I maintains two RPBs: RPB_A for node A and RPB_B for node B.

The size of both APB and RPB are set to the PROBE_BUFFER_SIZE. This buffer size can be calculated from \( \text{START\_UP\_TIME} \times \text{BROADCAST\_RATE} \) or \( 15 \times 1 = 15 \) probe messages as recommended in the original ETX test-bed experiment [DEC03]. Both buffers are served in a first-in-first-out manner. Once a buffer is fully filled up with probe messages, the oldest probe message in the buffer will be replaced with a new one. This allows nodes to maintain only the latest probe messages sent or received in the buffers.

When a node needs to derive a metric value (i.e. SETX metric value) for a link, the node first requests for the RPB (for simplicity, hereafter, when we say a ‘RPB’ or ‘APB’, we mean the content stored in the RPB or APB) from the corresponding neighbouring node. Once the RPB is received, the node compares the receiving RPB with its APB, and based on the comparison result, the node calculates the forward delivery ratio, on which it calculates the SETX metric value. The details of this process are described in the following section.

Figure 3.12: The use of APBs and RPBs in the SETX protocol
### 3.3.2 Detailed Protocol Description

<table>
<thead>
<tr>
<th>Neighbouring node (K)</th>
<th>Initiator node (I)</th>
<th>Neighbouring node (J)</th>
</tr>
</thead>
</table>

**Step 1: Generating Probe Messages and Probe Packets**

**Step 2: Advertising Probe Packets**

**Step 3: Receiving Probe Packets and Calculating receiving_rate**

**Step 4: Storing Probe Messages**

**Step 5: Exchanging RPBs**

**Step 6: Verifying Acknowledge Probe Messages**

**Step 7: Calculating SETX value**

---

Figure 3.13: Procedures of the SETX protocol

Figure 3.13 shows the process of how this mechanism can be used to find a SETX value between node I, J and K. It is explained in 7 steps: Generating Probe Messages and Probe Packets, Advertising Probe Packets, Receiving Probe Packets and Calculating receiving_rate, Storing Probe Messages, Exchanging Received Probe Buffers (RPBs), Verifying Acknowledgement Probe Messages and Calculating SETX value.

**Step 1: Generating Probe Messages and Probe Packets**

![Diagram of SETX Procedure Step 1, Generating Probe Packets]

**Figure 3.14: SETX Procedure Step 1, Generating Probe Packets**
Initiator \(^1\) \(I\) and neighbouring nodes \(J\) and \(K\), generate a probe packet denoted as \(p_I, p_J\) and \(p_K\) (shown in Figure 3.14). The packet format of a probe packet is shown in Figure 3.15. It contains the sender's IP address and a probe message \((pm)\). If IPv4 is used, the sender address field is 4 bytes or 32 bits long (i.e. the size of IPv4 address). However, if IPv6 is used, the size of this field will need to be modified to accommodate the IPv6 address. That is, the sender address field will need to be extended to 16 bytes long.

![Figure 3.15: Probe Packet Format](image)

The probe message field contains a probe message \((pm)\) that is 8 bytes long. A probe message is a random value that is generated from a random value generator. The random value generator should produce random values that are hard to predict. That is, given a value, \(k > 0\), and a sequence of value, \(n_1, n_2, \ldots, n_k\), an observer cannot predict \(n_k\) even if all of \(n_1, \ldots, n_{k-1}\) are known. Given the complete knowledge of the algorithm or hardware generating the sequence and all of the previous value, it must be computationally infeasible to predict what the next random value will be.

The question is why the probe message is chosen to be 8 bytes or 64-bits long. The size of a probe message must be long enough so that the random value generator unlikely give the same probe message (this is called a collision) within a period of time (e.g. at least within a year\(^2\)) is sufficiently small. Considering a short probe message like 24 bits (4-bytes) long,

---

1 An initiator node is the node that generates and broadcasts a probe packet to its neighbour nodes. Although other nodes also generate and broadcast their probes, the initiator node is the node that we focus on.

2 A low-power wireless sensor node can last more than a year with a single charge of 2 x AA battery. This sensor node has been implemented and described in [MAN11].
it would take just a few hours before the probability that we will generate the same probe message is high (say more than 0.5). This situation is similar to a birthday attack \[\text{BRI12}\], where \( P(\text{n}_\text{probe}\text{.generated}) \) is the probability of at least two of the \( n \) probe messages sharing the same value. However, it is easier to first calculate the probability \( \overline{P}(\text{n}_\text{probe}\text{.generated}) \) that all probe messages are different. For \( \overline{P}(\text{n}_\text{probe}\text{.generated}) \),

\[
\overline{P}(\text{n}_\text{probe}\text{.generated}) = \frac{n\text{\_\text{probe}\text{.generated}}! \times \binom{\text{total\_\text{probe\_event}}}{\text{n}_\text{probe}\text{.generated}}}{\text{probe\_size}^{\text{n}_\text{probe}\text{.generated}}}
\]

where \( n\text{\_\text{probe}\text{.generated}} \) is a number of probe messages generated. "!" is the fractional operator. \( \text{total\_\text{probe\_event}} \) is a total number of event that the value of probe message can be without having the same value (i.e. \( 2^{24} \) in this case). \( \binom{\text{total\_\text{probe\_event}}}{\text{n}_\text{probe}\text{.generated}} \) is a binomial coefficients, which

\[
\binom{\text{total\_\text{probe\_event}}}{\text{n}_\text{probe}\text{.generated}} = \frac{\text{total\_\text{probe\_event}}!}{\text{n}_\text{probe}\text{.generated}}! \times \left(\text{total\_\text{probe\_event}} - \text{n}_\text{probe}\text{.generated}\right)!}
\]

After \( \overline{P}(\text{n}_\text{probe}\text{.generated}) \) is found, then \( P(\text{n}_\text{probe}\text{.generated}) \) can be calculated from

\[
P(\text{n}_\text{probe}\text{.generated}) = 1 - \overline{P}(\text{n}_\text{probe}\text{.generated})
\]

Given that the initiator node generates a probe message with a rate of 1 packet per second. Within 1 hour, 3,600 (60 seconds x 60 minutes) probe messages will be generated. This means \( n\text{\_\text{probe}\text{.generated}} \) is 3,600 and then the \( \overline{P}(\text{n}_\text{probe}\text{.generated}) \), the probability that all the probe messages are different is,

\[
\overline{P}(\text{n}_\text{probe}\text{.generated}) = \frac{3600! \times \binom{2^{24}}{3600}}{(2^{24})^{3600}}
\]

Then the probability that there will be at least 2 probe messages with the same value, \( P(\text{n}_\text{probe}\text{.generated}) \), is

\[
P(\text{n}_\text{probe}\text{.generated}) = 1 - \frac{3600! \times \binom{2^{24}}{3600}}{(2^{24})^{3600}}
\]
\[ P(n_{\text{probe\_generated}}) = 0.3186 \]

From the probability of \( P(n_{\text{probe\_generated}}) \), there is around 30% chance that 2 out of 3,600 probe messages will be the same. In other words, a chance that a node will generate the same probe message with in 1 hour is around 30%. However, if there are 2 nodes in the network, the probability that two of the probe message will be the same will be double which is 70%. So this means, it is likely to have two probe messages with the same value within 1 hour.

Since using a small size of probe message can cause the collision problem very quickly, why not simply select a significantly larger size of a probe message, such as 256 bits (32 bytes) or 512 bits (64 bytes)? Surely, the bigger the size of a probe message, the better it is in terms of avoiding a collision, and the harder it is for a malicious node to guess a probe message correctly. Having a bigger probe message size will increase the size of a probe packet. This imposes a higher level of costs in terms of bandwidth and storage requirements, as more bandwidth will be required to transmit the probe packets, and also more memory space will be required to store them. However, with the size of the probe message increased, it is arguably that it adds a little cost when compare to the extra of security protection.

The question is how big the size of a probe message should be. With a 64-bit length, and assume that each node generates 1 probe message per second for a year (assuming that there are 365 days in a year) and there are 50 nodes in the network, then a probability that two probe messages will be the same is,

\[
P(n_{\text{probe\_generated}}) = 1 - \frac{(3600 \times 24 \times 365 \times 50)! \times 2^{64}}{(2^{64})^{3600 \times 24 \times 365}} = 0.0647454
\]

This means after 50 mobile nodes generating probe messages for a year, a chance that the two probe messages will be the same is around 6.5%. This \( P(n_{\text{probe\_generated}}) \) is for a network with 50 mobile nodes. However, if the network has less mobile nodes, for example, for 10 mobile nodes network, \( P(n_{\text{probe\_generated}}) \) will be even smaller which is around 0.27% This probability is considered as a very low chance that two or more probe
messages will have the same value. This 64-bit length is a compromise between security and costs, so is used as a benchmark in the simulation investigation presented in this thesis.

**Step 2: Advertising Probe Packets**

![Figure 3.16: SETX Procedure Step 2, Advertising Probe Packets](image)

Node $I$ broadcasts freshly generated probe packets at a given interval (governed by the $BROADCAST\_RATE$) to its neighbouring nodes (in this case they are nodes $J$ and $K$). In the mean time, nodes $J$ and $K$ also broadcast their respective probe packets as shown in Figure 3.16. This step continues until the battery of the node runs out, or until the node leaves the network.

The $BROADCAST\_RATE$ is agreed upon among the nodes in the network before the probe packet advertising phase can begin. The default $BROADCAST\_RATE$ is 1 packet per second. This value is recommended and tested by the test-bed experiment of the original ETX protocol [DEC03]. The $BROADCAST\_RATE$ can be modified depending on the requirement of the application. If the $BROADCAST\_RATE$ is higher, probe packets will be broadcast more frequently. This allows nodes to maintain fresher or more accurate routing metric value. So a higher $BROADCAST\_RATE$ is more preferable to a network with a higher level of mobility. Of course, a higher $BROADCAST\_RATE$ will introduce more traffic overhead into the network, and impose more processing loads on the nodes.

On the other hand, if the $BROADCAST\_RATE$ is low, less traffic overhead will be injected into the network and less resources (e.g. CPU processing time, battery consumption) will be consumed at the nodes. A lower $BROADCAST\_RATE$ is more suitable for a network with a lower level of mobility as the network topology does not change so often.
Step 3: Receiving Probe Packets and Calculating receiving_rate and reverse delivery ratio

Figure 3.17 shows this step. Node I calculates a receiving_rate for each individual neighbouring node. As mentioned in Section 2.3.3.3, the receiving_rate is the average rate of the number of probe packets the node has received within the last START_UP_TIME (second). For example, node J broadcasts probe packets at a rate of 1 packet per second. In the last 10 seconds, if node I receives only 7 probe packets, the receiving_rate of node I from node J would be 0.7 packet per second.

Given that the number of probe packets received within the last START_UP_TIME seconds is $n_{probe_{receive}}$. The receiving_rate can be calculated by Equation 3.1,

$$receiving\_rate = \frac{n_{probe_{receive}}}{START\_UP\_TIME \times BROADCAST\_RATE}$$ \hspace{1cm} \text{Equation 3.1}

After the receiving_rate is calculated, the reverse delivery ratio can be calculated by Equation 3.2,

$$reverse\_delivery\_ratio = \frac{receiving\_rate}{BROADCAST\_RATE}$$ \hspace{1cm} \text{Equation 3.2}

The reverse delivery ratio will be used to calculate SETX metric value later on.

In the case that receiving_rate is higher 1, this means node has received more packets than it is supposed to receive. This can happen when there is a malicious node sending probe packets with a higher than BROADCAST_RATE or there is an impersonation node broadcasting a packet with another node's identity. If this happens, a node that detects
these malicious activities should drop this link to avoid any further implications of the attacks.

**Step 4: Storing Probe Messages**

![Diagram showing probe messages storage](image)

**Figure 3.18: SETX Procedure Step 4, Storing Probe Message**

In this step, both probe messages which have been advertised and probe messages which receives from a neighbouring node will be stored into the corresponding probe buffers (shown in Figure 3.18). When initiator I advertises probe packets, it stores the advertised probe messages (which contained in the probe packets) into its APB_I. Node J and node K also store their advertised probe messages into their respective buffers, APB_J and APB_K. These probe messages will be used to verify an acknowledgement probe message received from the neighbouring nodes later on (in Step 5).

In the meantime, initiator I also receives probe packets advertised by nodes J and K. Here, the probe messages contained in probe packets received from node J will be stored in I’s RPB^_I_J. Similarly, the probe messages contained in probe packets received from node K will be stored in I’s RPB^_I_K. Nodes, J and K, also store the probe messages contained in probe packets received from their respective neighbours in the same way as initiator I. The probe messages contained in RPBs are now called *reply probe messages*. They will be sent to the respective neighbouring nodes, upon request, for the calculations of *forward delivery ratios* by the corresponding neighbouring nodes in Step 5.
Step 5: Exchanging RPBs

When initiator I wants to calculate a SETX metric value for a link to a neighbouring node J, initiator I will request node J to exchange each other’s RPBs as shown in Figure 3.19. That is, initiator I sends the corresponding $RBP^I_J$ to J and J sends $RBP^J_I$ to I. Similarly, initiator I will exchange its RPB with node K, and so on. These reply probe messages will be used to calculate the forward delivery ratios for the links concerned provided that the acknowledgement probe messages received pass the verification process. The process of verifying the probe messages is described in Step 6.

![Figure 3.19: SETX Procedure Step 5, Exchanging RPBs](image)

The implementation of the SETX protocol does not require the use of additional packets to transmit RPBs. RPBs can be carried inside routing control packets, e.g. carried in the RREQ and RREP packets of the AODV protocol. In this thesis, we use AODV as an example, so the RPBs are carried in the RREQ and RREP packets. This only extends the size of the routing packets. Figure 3.20 shows the RREQ and RREP packet formats used in the SETX protocol.

![Figure 3.20: Formats of RREQ and RREP packets used in SETX](image)
Step 6: Verifying Acknowledgement Probe Messages

Once the initiator I, receives the acknowledgement probe messages (i.e. $RPB_I$) from node $J$, initiator I verifies these probe messages by comparing the acknowledgement probe messages with those stored locally in the $APB_I$ buffer. Given that $x_i$ is the $i^{th}$ probe message stored in $APB_I$, $y_j$ is the $j^{th}$ probe message stored in $RPB_I$ and $n_{matched}$ is the number of probe messages that have been positively verified, the pseudo code of the probe verification algorithm is given below:

\[
n_{matched} = 0 \\
\text{for } i \text{ from } 0 \text{ to PROBE_BUFFER_SIZE step by 1} \\
\quad \text{for } j \text{ from } 0 \text{ to PROBE_BUFFER_SIZE step by 1} \\
\quad \quad \text{if (} x_i \text{ equal } y_j \text{) then} \\
\quad \quad \quad n_{matched} = n_{matched} + 1 \\
\quad \quad \quad \text{exit for}
\]

Figure 3.21: Probe Verification Algorithm

The output from the algorithm is $n_{matched}$ which indicates the number of valid probes that the neighbouring node can show to the initiator node. With the value of $n_{matched}$, the initiator calculates the forward delivery ratio using the following equation,

\[
\text{forward delivery ratio} = \frac{n_{matched}}{n_{APB}}, n_{APB} > 0 
\]

Equation 3.3

where $n_{APB}$ is the number of probes stored in APB and $0 < n_{APB} \leq PROBE_BUFFER_SIZE$.

Step 7: Calculating SETX value

Once both the reverse delivery ratio (from Step 3) and the forward delivery ratio are calculated, the initiator uses Equation 3.4 to calculate the SETX value. This is the same equation used in the original ETX protocol.

\[
SETX = \frac{1}{(\text{reverse delivery ratio} \times \text{forward delivery ratio})} 
\]

Equation 3.4
Using this SETX protocol, neighbouring nodes have to send all probe messages they receive back to the corresponding initiator node. These returned probe messages (i.e., acknowledgement probe messages) serve as the evidence of the quality of the link concerned. As the probes contain random values, if a neighbouring node has not received a valid probe, but would like to forge one, it would have to guess a random value that matches with one of those stored in the initiator’s APB buffer, and the chance for this being successful is very small. This probability will be discussed in Section 3.4.2.1.

To conclude, unlike the original ETX protocol [DEC03] where the reverse delivery ratio of a link is calculated by the initiator, and the forward delivery ratio of the link is calculated and notified by the other (neighbouring) node of the link, the SETX protocol, described above, allows the initiator to calculate both reverse delivery and forward delivery ratios. To prevent potential forgery of the acknowledgement probes returned by the neighbouring node, SETX requires that all the probes received be verified against the original copy maintained by the initiator node. Based on the verification outcome, the initiator node calculates the delivery ratios itself.

The SETX protocol does not require additional packets to be generated, rather it uses existing routing control packets to transport the probes, thus keeping the overheads low. So the major additional traffic overhead of the SETX protocol over the ETX protocol is the extended length of a routing packet. The additional length per routing packet is \(\text{PROBE_BUFFER_SIZE} \times 8\) bytes (to carry a \(\text{RPB}\)). In addition, this happens only when a RREQ and a RREP packet are dispatched. The detailed investigation of the impact on performance by this increase in a routing packet size is studied using simulation. The simulation results are presented in Section 3.6.

### 3.3.3 Probe Message Protection using Cryptographic Techniques (An Optional Solution)

Another approach to impersonation attacks is to use a cryptographically generated digital token such as a digital signature or a keyed hash value. These tokens can provide authenticity protection to a probe ensuring that any forged probes, or any unauthorised alterations to authentic probes can be detected.
The SETX protocol does not use this technique by default (though the protocol can easily be extended to support the use of this technique). The reason for discussing this approach here is to show that there is an optional approach to the problem addressed by the SETX protocol, but this optional approach has some usability problems which can make its deployment in a dynamic MANET environment costly.

### 3.3.3.1 Digital Signature Technique

A digital signature is an electronic signature used to authenticate the identity of the sender of a message. It can also be used to detect whether the original content of a message or document has been tampered with during transmission. In other words, it can provide data integrity, data authenticity and non-repudiation services. It is often used in situations where there is a lack of trust between senders and receivers as it cannot be imitated by any other nodes.

With this signature technique, an initiator node signs a probe packet and attaches the digital signature with the probe packet before sending it to the receiver. Once a neighbouring node (i.e. the receiver) receives the probe packet along with its signature, it verifies the authenticity of the digital signature. If the verification is positive, it stores a probe message contained in the probe packet into its RPB. Otherwise, the probe packet is discarded.

As shown in Figure 3.22, given a probe message, $pm_I$, and an initiator’s identity ($I$), the hash value, $h$, is generated from $pm_I$ and $I$ using $I$’s private key, $KR_I$. Then the cipher text ($E_{KR_I}[h]$) (i.e. $I$’s digital signature on this probe packet$^1$) is attached to the probe packet before being sent to the receiver. To verify the signature, the receiver uses $I$’s public key, $KU_I$, to recover the hash value in $E_{KR_I}[h]$, and compares this value with a hash value ($h^*$) freshly generated from $pm_I$ and $I$. If the two hash values are equal, the authenticity of the message and non-repudiation of its origin are verified as belonged to initiator $I$.

---

$^1$ In this case, the probe packet contains $I$’s identity ($I$), probe message ($pm_I$) and $I$’s digital signature ($E_{KR_I}[h]$).
3.3.3.2 Keyed Hash Function Technique

A probe packet may also be protected by using a symmetric key algorithm, e.g. a keyed hash function. A keyed hash function can be up to 100 times faster in software implementation or 1,000 times faster in a specialised hardware implementation [OMA97]. Especially in MANETs where mobile nodes are low in performance and have limited resources, the use of digital signatures can significantly reduce the performance of the underlying network operations. The keyed hash function becomes a more attractive option to data protections in MANETs.

However, as probe packets are advertised by broadcasting them through a wireless channel, there is an issue as how symmetric keys should be managed, i.e. (1) who should generate them, (2) how should they be distributed to their intended recipients, (3) how should they be stored so that it is hard to access them by unauthorised entities, and (4) how they should be shared. Answers to these questions are not straightforward. For example, for question (4), there are two possible ways by which symmetric keys may be shared. One is for a group of nodes to share a group secret key (i.e. use group key sharing), and the other is for each pair of nodes to share a secret key (i.e. use pair-wise secret key sharing).

If we use the group key sharing method, all the nodes in this group will have the knowledge of this secret key. All the probes generated by any member of this group will be protected by this key (the key is used to generate a keyed hash value for each of the probes). This method is suited to the case where the members of the group trust each other. It cannot be used to protect impersonation attacks by a malicious insider (i.e. a node from
the same group). This is because all nodes in the group share the same key. If a malicious node has the key, it can simply forge a probe packet.

If we use the pair-wise secret key sharing method, each pair of nodes needs to have a unique secret key which is only known to the two nodes. For example, $K_{IA}$ is the secret key shared between nodes $I$ and $A$. Apart from these two nodes, no other node should have any knowledge of this key. However, this method may require each node to obtain secret keys for each of all other nodes in the network. If there are $n$ nodes in the network, each node will have to maintain $n-1$ secret keys. This means there could be $n^2$ unique secret keys in the network.

Assume that initiator $I$ have two neighbours, node $A$ and node $B$, and also assume that the keys the initiator shares with each of the neighbours are $K_{IA}$ and $K_{IB}$. To broadcast a probe packet, $p_I$, to these two nodes, initiator $I$ generates two hash values, one (i.e. $H(K_{IA}, p_I)$) for $A$ and the other (i.e.$H(K_{IB}, p_I)$) for $B$, using their respective keys. These hash values, along with the recipients’ identities, are appended to the probe message and the initiator’s identity, before being broadcast. This process is shown in Figure 3.23.

**Initiator $I$ generates a probe packet**

```
Diagram:
I
\[ \text{pm}_I \]

\[ H(K_{IA}, \text{pm}_I) \]
\[ K_{IA} \]

\[ A \]
\[ H(K_{IA}, \text{pm}_I) \]

\[ H(K_{IB}, \text{pm}_I) \]
\[ K_{IB} \]

\[ B \]
\[ H(K_{IB}, \text{pm}_I) \]

\[ \text{Hash Function} \]
\[ I \]
\[ \text{Node I’s identity} \]
\[ K_{IA} \]
\[ \text{Secret key shared between I and A} \]
```

**Figure 3.23:** Initiator $I$ generates a probe packet for neighbouring node $A$ and node $B$ using a secret key technique

When the message arrives, node $A$ only extracts its corresponding part of the packet i.e. $pm_I, A$ and $H(K_{IA}, pm_I)$, while node $B$ also extracts only its corresponding part, i.e. $pm_I, B$ and $H(K_{IB}, pm_I)$. Here, value “$A$” indicates that the hash value in the next field (i.e. $H(K_{IA}$,
Node A verifies a probe packet

![Diagram](image)

This pair-wise secret key sharing method is more secure than the group key sharing method, as a pair-wise shared key is only known to two nodes, so easier to detect if the key is abused. However, with this method, the number of keys that an initiator needs to manage, and the number of hash values that it generates, are dependent on the number of neighbours the initiator has. The more neighbours the initiator has, the more overhead costs it will introduce, in terms of key management and hash value generations and verifications.
For both methods, it is impractical in reality because the symmetric keys need to be distributed to the authorised entities before protections can be applied. This implies that an initiator node needs to know the list of the receivers before generating any probe packets.

3.4 Security Analysis of the SETX Protocol

This section performs a security analysis of the SETX protocol. In section 3.4.1, we discuss how the SETX protocol handles the security attacks we have identified on the ETX protocol, i.e. those discussed in Section 3.2. In section 3.4.2, we identify new security attacks on the SETX protocol.

3.4.1 Security Analysis against Security Attacks on ETX

3.4.1.1 Attacks on ETX Case 1: Advertising a falsified forward delivery ratio

In the ETX protocol, a neighbouring node is allowed to calculate and advertise a forward delivery ratio freely. This mechanism makes the ETX protocol vulnerable to falsifying forward delivery ratio attack, i.e. if the neighbouring node is malicious, then it can advertising a false forward delivery ratio when an initiator node requests one.

The SETX protocol solves this problem by preventing neighbouring nodes to calculate the forward delivery ratio themselves. It requires the neighbouring node to record probe messages broadcast by the initiator and send the newest set of probe messages they have received back to the initiator, and lets the initiator to calculate the forward delivery ratio. As the probe messages contain random values, to successfully forge a forward delivery ratio, the neighbouring node would have to return sufficient number of probe messages with the random values matching with those expected by the initiator, thus making the forgery of the forward delivery ratio harder. Section 3.4.2.1 gives a quantitative analysis of how hard (measured in terms of probability) this is to the neighbouring node.

3.4.1.2 Attacks on ETX Case 2: Modifying BROADCAST_RATE

As mentioned in Section 3.2.2, there are two types of attacks in this category, selfish attack and flooding attack.
Attacks on ETX Case 2.1: Selfish Attacks

The (malicious) neighbouring node broadcasts probe packets at a rate that is lower than the agreed rate, or skips broadcasting some probe packets. In this case, the initiator will receive fewer probe packets than expected, which means a lower receiving_rate. Since reversing delivery ratio is calculated as \( \frac{\text{receiving_rate}}{\text{BROADCAST_RATE}} \), the reversing delivery ratio will be decreased too. As a result, the calculated ETX value will be lower than the actual value. In other words, like the ETX protocol, the SETX protocol is also vulnerable to this type of attacks.

One possible way to detect this type of attack is by monitoring the probe packet arriving rate to see if it is lower than BROADCAST_RATE. However, as the probe packet arrival rate may also be affected by other network or channel conditions, such as network congestions and/or radio interferences, detecting selfish attacks by means of detecting the probe packet arrival rate can be misleading.

Attacks on ETX Case 2.2: Flooding Attacks

Alternatively, a neighbouring node may attempt to forge an ETX value by broadcasting probe packets at a rate higher than the agreed. Unlike the ETX protocol, the SETX protocol has a simple mechanism to detect this type of attacks. The detection mechanism is discussed in Step 3 of SETX process (Section 3.3.2). That is, the initiator checks whether the receiving_rate is higher than the BROADCAST_RATE (i.e. receiving delivery ratio > 1). If it is higher, then there is a chance that this neighbouring node is a malicious node, so the initiator can exclude this node from the route selection process. However, this technique has its limitations. If the network and channel conditions are not good, the receiving_rate may not exceed the BROADCAST_RATE. In this case, the initiator may not be able to detect the attack. However, if the attack is caused by impersonation attacks by other nodes (rather than by increasing the broadcast rate by the neighbouring node itself) (as mentioned in Section 3.2.3), probe messages protection techniques (discussed in Section 3.3.3) may be applied to counter such attacks.
3.4.1.3 Attacks on ETX Case 3: Impersonation Attacks (Active Selfish attacks)

In the ETX protocol, the recipient of a probe packet does not verify if the probe packet received is legitimate or not. A neighbouring node (i.e. the recipient of the probe packets sent by the initiator) always believes that the probe packets are from the initiator. However, a selfish (third party) node may broadcast probe packets in the name of the initiator in an attempt to boost the ETX value of the link linking the initiator and the intended recipient of the probe packets to ensure that the selfish node is not selected en-route. In this case, the neighbouring node will receive more probes than it should have (as probes are coming from both the initiator and the malicious/selfish node). As a result, the ETX metric value of a link between the initiator node and this neighbouring node will be artificially boosted.

However, as to be discussed in the section below, with the SETX protocol, it is harder for another node to carry out this attack successfully. This is because, if a malicious node impersonates the initiator to advertise probe packets using the initiator node’s identity. The false probe messages contained in the false probe packets will be received, stored and later forwarded to the initiator. The initiator will verify these false probe messages against its own record of the probe messages sent. Owing to the interleaving effect of the false probe messages (sent by the malicious node) and the genuine probe messages (sent by the initiator itself), the matching probe count may be smaller than the count without this attack. In other words, in the SETX protocol, this attack may make the resulting ETX value worse – the opposite from what the malicious node would be hoping for. The outcome of the impersonation attack on the SETX protocol is further discussed in details in Section 3.4.2.3.

3.4.2 Security Analysis against Security Attacks on SETX

3.4.2.1 Attacks on SETX Case 1: Probe Message Guessing Attacks

Malicious node $M$ misses some probe messages from initiator $I$, but it wants to be selected en-route. Node $M$ cannot simply advertise a falsified forward delivery ratio as in ETX to
let initiator \( I \) believes that it has the best route to the destination node. Node \( M \) will need to obtain the missing probe message instead. One way to obtain a probe is to guess or generate a probe message by itself. Here, node \( M \) tries to guess a content of the missing probe message. The procedure of the attack is as follows.

**Figure 3.25: Probe Message Guessing Attacks — Step 1**

**Step 1:** Initiator \( I \) broadcasts its probe packets \( (p_I) \) to its neighbouring nodes, including malicious node \( M \) and neighbouring node \( A \) (shown in Figure 3.25). In the meantime, nodes \( M \) and node \( A \) also advertise their probe packets to initiator \( I \) too (i.e. \( p_M \) and \( p_A \) respectively). Unfortunately, node \( M \) does not receive a probe packet \( p_I \) from initiator \( I \). This may be because initiator \( I \) has moved away from node \( M \), or the probe packet is corrupted by a wireless interference issue.

**Figure 3.26: Probe Message Guessing Attacks — Step 2**

**Step 2:** As node \( M \) did not receive a probe packet \( p_I \) from initiator \( I \) (in Step 1), node \( M \) would not receive the probe message \( pm_I \) from \( I \) either. In this case, node \( M \) wants to falsify the forward delivery ratio by trying to get \( pm_I \). Node \( M \) guesses the content of \( pm_I \) and stores it into its RPB. When initiator \( I \) requests the RPBs from its neighbouring nodes, node \( A \) will send a truthful \( RPB^I_A \), but node \( M \) sends the falsified \( RPB (RBP^I_M) \) back to initiator \( I \) as shown in Figure 3.26.
Step 3: initiator I now has both RPBs from node A and node M. Initiator I then calculates the SETX metric values for both links (i.e. I→A and I→M) (shown in Figure 3.27). The calculated SETX metric value of the link I→A is truthful, as node A is an honest node. However, the SETX metric value of the link I→M may not be truthful. This is because node M sends a fabricated RPB*_{I→M} to initiator I. The value of SETX will be falsified if node M could guess the content of the missing probe message pm_{I→M} correctly. The question is how difficult it is for node M to guess the correct probe message to falsify the metric value.

In the worst-case scenario (the worst case for the initiator), node M misses only 1 probe message from the last 15 probe messages. To correctly guess 1 probe message, node M has only 1 in 2^{64} chance (or the probability of 2^{-64}). This is because 1 probe message is 64 bits long, and each bit has 2 possibilities: either 0 or 1. So to guess 64 bits right, the probability will be 2^{-64}.

On the other hand, in the best-case scenario (the best for the initiator), node M does not receive any probe message for the last 15 probe messages. Node M has to guess all the missing 15 probe messages correctly. The probability would be 1 in 2^{64} \cdot 15 or (2^{-64} \cdot 15). See Appendix A for more details on the probabilities.

Node M has only one chance/opportunity to guess the missing probes correctly. There is no feedback from initiator I to inform node M that the guessed probe is correct or wrong. After node M submits its RPB to initiator I, node M cannot resubmit the RPB again. So
Brute Force Attacks\(^1\) are not possible for node \(M\) to keep on trying to find a correct probe message in the SETX protocol.

To summarise, given the 64-bit length of the probe payload, the chance for node \(M\) to successfully guess a probe value is very small. If the malicious node does have such a hit, and, in addition, if the ETX values for other links are not better than the ETX value forged by the malicious node, this malicious node may be chosen en-route by the initiator.

### 3.4.2.2 Attacks on SETX Case 2: Cooperative Black hole Attacks

Malicious node \(M_1\) wants to be included en-route. Node \(M_1\) has a cooperative black hole node \(M_2\) helping it to forge the SETX metric value. Here, node \(M_1\) does not receive a probe message \(pm_I\) contained in a probe packet \(p_I\) broadcast by the initiator \(I\), but node \(M_2\) has received this probe packet \(p_I\). Then node \(M_1\) can ask node \(M_2\) to send it the missing probe message \(pm_I\) to it. When initiator \(I\) requests for a \(RPB\), node \(M_1\) replies with its \(RPB\) which contains the probe messages \(pm_I\) which node \(M_1\) did not receive. Initiator \(I\) then verifies the probes in the \(RPB\). All the probe messages (including \(pm_I\)) will pass the verification. As a result, the forward delivery ratio is modified (higher than the truthful value). The detailed process of this attack is as follows.

*Figure 3.28: Cooperative Black Hole Attacks – Step 1*

---

\(^1\) Brute Force Attack is a mechanism to search for a correct answer by trying all possibilities until it finds the correct answer. It may need a lot of attempts before the correct answer can be found. In our problem context, this attack is not possible as an attacker only has 1 attempt to guess a probe correctly.
Step 1: Initiator I broadcasts a set of probe packets (including $p_I$) to malicious node $M_1$ and malicious node $M_2$. In the meantime, nodes $M_1$ and $M_2$ also advertise their respective probe packets to initiator I too (e.g. $p_M$ and $p_A$). But somehow node $M_i$ fails to receive one of the probe packets sent by initiator I, $p_I$. As a result, node $M_1$ did not receive probe message $pm_I$ which is contained in the probe packet $p_I$.

![Diagram of Step 2: Cooperative Black Hole Attacks](image1)

**Figure 3.29: Cooperative Black Hole Attacks — Step 2**

Step 2: node $M_1$ requests the missing probe message $pm_I$ from node $M_2$. This can be done without the knowledge of initiator I. Node $M_1$ may send a request to node $M_2$ using a tunnel (encrypted channel).

![Diagram of Step 3: Cooperative Black Hole Attacks](image2)

**Figure 3.30: Cooperative Black Hole Attacks — Step 3**

Step 3: node $M_2$ sends the requested probe message $pm_I$ to node $M_1$. Now node $M_i$ has obtained the missing probe message $pm_I$. Similar to the previous step, node $M_2$ may send the requested probe message $pm_I$ through the same encrypted tunnel. Thus, the initiator I will not be aware of this activity.
Step 4: when initiator $I$ requests a RPB from node $M_1$. Node $M_1$ will send its RPB back to initiator $I$. This RPB will include the missing probe message $pm_I$ which was received from node $M_2$.

Step 5: initiator $I$ calculates the forward delivery ratio, and then the SETX metric value using RPB receives from node $M_1$. Here, the RPB received from node $M_1$ is not truthful since it contains a missing message $pm_I$ received from node $M_2$. However, since the probe message $pm_I$ is correct (node $M_2$ received it from initiator $I$), it will pass the verification process. As a result, the forward delivery ratio of the link $I \leftrightarrow M$ will be modified.

By collaborations, black hole nodes can help each other to collect missing probe messages and to fill up their RPBs. The number of cooperative black hole nodes is not limited to 2. There can be as many nodes as the attacker wants. However, there is one condition to this type of attacks, that is, the missing probe message(s) at one of the black hole nodes must be received by at least one of the other cooperative black hole nodes.

For example, as shown in Figure 3.33, there are 4 malicious nodes, $M_1$, $M_2$, $M_3$ and $M_4$. Initiator $I$ advertises probe packets: $p_{I1}$, $p_{I2}$, $p_{I3}$ and $p_{I4}$. Node $M_1$ may have only received
the first probe packet $p_{I1}$, Node $M_2$ only received the second probe packet $p_{I2}$, Node $M_3$ only received the probe packet $p_{I3}$ and node $M_4$ only received probe packet $p_{I4}$. In other words, each node may have received only 1 probe message from initiator $I$ as they have received only 1 probe packet. However, the four nodes can exchange the received probe message with each other. Figure 3.34 shows the result after all the cooperative black hole nodes exchange their probe messages. As a result, all nodes can obtain all the probe messages: $p_{I1}, p_{I2}, p_{I3}$ and $p_{I4}$ sent by initiator $I$.

![Figure 3.33: Cooperative Black Hole nodes $M_1$ to $M_4$ receiving only 1 probe message each](image)

![Figure 3.34: Cooperative Black Hole nodes $M_1$ to $M_4$ exchanging their probe messages with each other](image)

The SETX protocol cannot detect, nor thwart this cooperative black hole attack. As the nature of wireless communication, all receivers within the transmission range can receive the same data. Even the initiator node generates an individual probe message tailoring for each individual neighbouring node. As long as one cooperative black hole node can receive probe messages from the initiator, they will be able to share the probe messages with each others.
3.4.2.3 Attacks on SETX Case 3: Impersonation Attacks (Black Hole Attacks)

A malicious node \( M \) impersonates an initiator \( I \) in order to modify the *forward delivery ratio* of a link connecting the initiator and one of its neighbours. Node \( M \) advertises probe packets with initiator \( I \)'s identity to a victim node \( A \). The victim node thinks the probe packets are from the initiator, so stores the probe messages contained in the probe packets in its \( RPB \). These impersonated probe messages will be interleaved with those (genuine probe messages) sent by the initiator. In other words, the impersonated probe messages will pollute the probe messages from the initiator. When initiator \( I \) request for the \( RPB \), node \( A \) replies initiator \( I \) with its \( RPB \) containing the impersonated probe messages. The probe messages will fail the verification process. As a result, the *forward delivery ratio* of the link between initiator \( I \) and the victim node \( A \) (\( I \leftrightarrow A \)) will be reduced. So the SETX metric value of the link becomes worse than the actual value. As the SETX metric value of the link \( I \leftrightarrow M \) has not been falsified, the chance that initiator \( I \) will select node \( M \) en-route is higher. The detail of the attack procedure is as follows.

![Figure 3.35: Impersonation Attacks on SETX – Step 1](image)

Step 1: Initiator \( I \) broadcasts probe packets, \( p_{I1}, p_{I2}, p_{I3} \). Node \( A \) receives the probe packets from initiator \( I \), so it stores the probe messages, \( pm_{I1}, pm_{I2}, pm_{I3} \) that received from \( I \) into its \( RPB \).
Step 2: Node $M$ impersonates initiator $I$ by advertising probe packet $p_I^*$ using $I$’s identity. Node $A$ believes that $p_I^*$ is from initiator $I$, so it stores the probe message $pm_I^*$ into its $RPB$. Now node $A$’s $RPB$ contains both legitimate probe messages (i.e. those from $I$) and false (that from $M$) probes: $pm_{I1}, pm_{I2}, pm_{I3}$ and $pm_I^*$.

Step 3: Initiator $I$ continues to advertise its probe packets, $p_{I1}$ to $p_{I10}$. Now node $A$ has received probe packets $p_{I1}$ to $p_{I10}$ from initiator $I$, and $p_I^*$ from malicious node $M$. Assuming that $A$’s $PROBE_BUFFER_SIZE$ is 10, but node $A$ has now received one probe message more than $PROBE_BUFFER_SIZE$ (i.e. it has received 11 probe messages so far). As node $A$ only keeps the newest 10 probes in its $RPB$s, $pm_{I1}$ will be replaced with $pm_{I10}$. A’s $RPB_I$ is shown as in Figure 3.37.
Step 4: Initiator node $I$ requests $RPB$ from node $A$. Node $A$ sends the $RPB^{*I_A}$ to initiator $I$.

Step 5: Now initiator $I$ calculates the SETX value. It compares $APB$ with the received $RPB$. As we can see, the initiator node can match all probe messages from its $APB$ with the $A$’s $RPB$ except $pm_{I1}$ and $pm_{I*}$. This is because $pm_{I1}$ has been replaced with $pm_{I10}$. As a result, the forward delivery ratio ($d_{I\rightarrow A}$) is decreased.

In the ETX protocol, these impersonation and flooding attacks will lead to a better ETX metric value, as the neighbouring node will receive more probes than those broadcast by initiator $I$. However, the effect of these attacks on the SETX protocol is just the opposite. They will make the SETX metric value worst. This is because the false probes will fail the probe verification at the initiator node but they do occupy the buffer space causing legitimate probes being discarded.
### 3.4.3 Discussion of the Security Analysis

Table 3.2 summarises the identified security attacks on ETX and on SETX protocols.

<table>
<thead>
<tr>
<th>Cases</th>
<th>Attack Types</th>
<th>ETX Forward Delivery Ratio</th>
<th>ETX Reversing Delivery Ratio</th>
<th>SETX Forward Delivery Ratio</th>
<th>SETX Reversing Delivery Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETX 1</td>
<td>Advertising a falsified</td>
<td>Vulnerable(^1)</td>
<td>n/a</td>
<td>Not Vulnerable(^2)</td>
<td>n/a</td>
</tr>
<tr>
<td></td>
<td>forward delivery ratio value</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ETX 2.1</td>
<td>Lower BROADCAST_RATE</td>
<td>n/a</td>
<td>Vulnerable</td>
<td>n/a</td>
<td>Vulnerable with Possible Solutions(^3)</td>
</tr>
<tr>
<td>ETX 2.2</td>
<td>Higher BROADCAST_RATE</td>
<td>n/a</td>
<td>Vulnerable</td>
<td>n/a</td>
<td>Partially Detected(^4)</td>
</tr>
<tr>
<td>ETX 3</td>
<td>Impersonation attacks</td>
<td>Vulnerable(^5)</td>
<td>n/a</td>
<td>Not Vulnerable(^6)</td>
<td>n/a</td>
</tr>
<tr>
<td></td>
<td>(Selfish)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SETX 1</td>
<td>Probe Guessing Attacks</td>
<td>n/a</td>
<td>n/a</td>
<td>Low probability to guess(^7)</td>
<td>n/a</td>
</tr>
<tr>
<td>SETX 2</td>
<td>Cooperative Black hole Attacks</td>
<td>n/a</td>
<td>n/a</td>
<td>Vulnerable(^8)</td>
<td>n/a</td>
</tr>
<tr>
<td>SETX 3</td>
<td>Impersonation attacks</td>
<td>n/a</td>
<td>n/a</td>
<td>Vulnerable with Possible Solutions(^9)</td>
<td>n/a</td>
</tr>
<tr>
<td></td>
<td>(Black hole)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3.2: The comparison list of attacks on ETX and SETX

[Note: n/a denotes that the attack is not applicable.
1. In ETX, a forward delivery ratio can be falsified by a malicious node. Instead of advertising a legitimate value, the malicious node can advertise a falsified value. This attack has been discussed in Section 3.2.1.
2. In SETX, a forward delivery ratio cannot be simply generated by a malicious node. The malicious node needs to obtain the relevant probe messages and send them back to the initiator node. Then the initiator node will calculate a forward delivery ratio by itself. This process is done by the SETX protocol. This has been discussed in Section 3.4.1.1.
3. This is a selfish attack. A technique that can be used to counteract this attack is to use a trust model [XIE08]. A trust model has been discussed in Section 2.3.3.4.
4. A malicious node can change its BROADCAST_RATE. This can only be detected when the receiving rate is higher than BROADCAST_RATE (i.e. reverse delivery ratio is more than 1). See Section 3.2.2 for more details.
5. In ETX, selfish nodes can impersonate an initiator by advertising a probe packet using the initiator’s identity. This will make the neighbouring node have a higher receiving_rate. As a result, the link between the initiator node and the neighbouring node will have a better ETX value. See Section 3.2.3 for more details.
6. In SETX, selfish nodes cannot inflate a SETX value of a link between the initiator node and a neighbouring node. This is because the impersonated probe message will fail the probe message verification. When a probe message fails the verification, the forward delivery ratio will be reduced. This attack has been discussed in Sections 3.4.2.3.
7. If a malicious node wants to modify a SETX metric value, it has to successfully guess the content of a probe message. In the worst case, a malicious node misses 1 probe message, the probability that it will guess this probe message correctly is \(2^{-64}\). In the best case, a malicious node misses all the 15 probe messages, the probability that it will guess all the probe messages correctly is \(2^{-64 \times 15}\). Section 3.4.2.1 has discussed this in detail.
8. Since probe packets are broadcast through wireless, all cooperative black hole nodes around the initiator node can receive the probe packets. If one of the cooperative black hole nodes receives these probe packets, they can then share probe messages contained in the probe packets with each other. There is no mechanism to detect this type of attacks, so SETX is vulnerable to these attacks.]
9. The impersonation attack can be detected by using a probe protection technique as mentioned in Section 3.3.3. For example, the initiator node may sign each probe packet with its digital signature before broadcasting them. All neighbouring nodes verify these probe packets. If the verification is successful, the probe packet will be accepted, otherwise, it will be discarded. In this way, malicious nodes will not be able to modify the metric value using this attack.

From Table 3.2, it can be seen that the SETX protocol is more secure than the ETX protocol in terms of countering black hole attacks by falsifying forward delivery ratios. However, SETX is not an all-in-one solution to prevent or thwart all kinds of security attacks. For example, it cannot protect the network against \textit{BROADCAST\_RATE} modification attacks and is not effective in protecting against impersonation attacks. To protect against these attacks, the SETX protocol will need to be extended with additional techniques (e.g. probe message protection techniques, see Section 3.3.3).

In addition, the SETX protocol also introduces new types of security attacks. These are probe guessing attacks, cooperative black hole attacks and impersonation attacks. The longer the probe length, the harder it is to successfully guess a probe. Of course, using a longer probe message will introduce more overhead, and consumes more bandwidth. That is, there is a trade-off between security and performance. The impersonation attack can be detected using a probe message protection technique.

3.5 Comparing the SETX Protocol with Related Works

This section compares our SETX protocol with related solutions. It first reviews the related solutions on countering black hole attacks in MANET routing protocols. Since our proposed SETX protocol is designed as a countermeasure to black hole attacks, our related work review focuses on solutions against black hole attacks in MANETs.

3.5.1 Related Works

Deng’s Routing Information Verification Scheme [DEN02]

Deng et al. [DEN02] addresses the black hole attack by modifying the AODV protocol. The method requires each node to verify whether an advertised route exists. This requires
each intermediate node along a route to append the address of the next hop node in AODV route request and route reply packets.

![Diagram of routing nodes]

**Figure 3.40: The example of Routing Information Verification Scheme [DEN02]**

This method can be explained by using an exemplar network topology shown in Figure 3.40. Here, nodes S and B have node A and C as their neighbouring nodes. Once node S receives the address of the next hop node (which is node B’s address) from a neighbouring node A (this is indicated as step (1) in the Figure), node S sends a verification packet to the next hop node to verify the existence of the next hop node and the routing metric value (i.e. the hop count) with the next hop node (2). This step may be done via another available routes, i.e. it may not be via node A. Upon the receipt of the verification packet, the next hop node of the neighbour node replies with another verification packet back to node S to confirm its existence and the metric value (3). If node S does not receive any reply, or if the routing information returned does not match with the one advertised by node A, then this neighbouring node A is assumed to be a malicious node. This approach can detect any non-existing routes falsely advertised by malicious nodes. However, the method is vulnerable to cooperative black hole attacks [AGR08, RAM03]. If both the neighbouring node and the next hop node are black hole nodes, the next hop node can respond to the source node with falsified routing information.

**Time-based Threshold Detection Scheme [TAM07]**

Tamilselvan’s solution is an extension of the original AODV protocol. The idea is to reduce the chance of selecting a black hole node en-route by waiting and checking for the replies from all the neighbouring nodes so as to find a safer route. Once the source node receives the first RREP packet, it sets a timer in the “TimeExpiredTable” so from this

100
moment on, the source node wait for other RREPs from other nodes. For each RREP packet received, it records the sequence number and the time when the RREP packet is received within a timeout value contained in Collect Route Reply Table (CRRT). After the timeout, the source node checks the records in CRRT to see whether there is any repeated next hop node (Two routes have the same intermediate node). If there is, the protocol assumes that the paths are correct or the chance of malicious paths is limited. The simulation study presented in [TAM07] shows that, in comparison with the original AODV protocol, this solution can achieve a higher packet delivery ratio with very little delay and overhead.

However, always assuming that a route which shares the same intermediate node with another route is legitimate might not always be efficient. There is also a case that the legitimate route does not share an intermediate node with another route. If this happens, the legitimate route can be suspected and discarded. If this legitimate route had a better routing performance than the other route, a worse performance route could be selected.

**Random Two-hop Acknowledge and Bayesian Detection Scheme [DJE08]**

Djenouri et al. proposed a solution to counter black hole attacks in MANETs. The solution can be explained in three phases; (1) monitoring (the neighbouring nodes), (2) detecting (black hole nodes using a Bayesian approach) and (3) isolating the black hole node.

1. **Monitoring:** it uses a watchdog scheme [MAR00] to detect whether a neighbouring node has actually forwarded packets on or simply dropped them. The idea is that once an intermediate node receives a packet, it needs to send a two-hop ACK (i.e. two-hop acknowledge) back to the two-hop upstream intermediate node. For example, if there is a route of A→B→C, then C has to send the two-hop ACK back to A. This allows A to check whether or not B has actually sent the packet to C. To prevent B falsifying the ACK from C, this phase requires the use of asymmetric cryptography (according to [DJE08]) to protect the authenticity of the ACK packet.

2. **Detecting:** the missing/dropping of a packet may not be caused by a malicious action; it could be caused by collisions or node’s mobility. At this phase, it allows nodes to
decide the behaviour of other nodes. The approach gives a high reputation to a well-behaving node, but reduces the reputation of a node with an unintentional or intentional packet dropping behaviour. The mechanism used to assign the reputation to a neighbouring node is similar to the Trust-based Routing Protocol [XUE04] which has been discussed in Section 2.3.3.4

(3) \textit{Isolating}: after a malicious node has been detected, all the witness nodes must discuss and jointly make a decision as whether or not to isolate this node. If all the witness nodes are in agreement, the malicious node will be isolated and excluded from any further network activities.

The simulation studies in [DJE08] show that the solution can achieve a lower false detection rate and a higher true detection rate than using the simple watchdog scheme [MAR00]. The performance of the network is improved if a malicious node is detected correctly. However, the drawback of the solution is that it cannot prevent cooperative black hole attacks. The cooperative malicious nodes may help each other to deceive the detection node using false information, or once a malicious node is detected, another malicious node may disagree to isolate the misbehaviour node.

The solutions described above are designed to counter black hole attacks on the AODV protocol. They employ a common approach, they use acknowledge packets or packet verification to check whether a route to a destination node exists or is valid. They cannot check the quality of the route which is what the ETX protocol tries to measure. Black hole nodes may fabricate the quality of a link and these methods cannot detect the attacks by fabrication. In other words, they cannot be used to counter black hole attacks on the ETX protocol. The only exception is Shila’s Algorithm to be discussed next.

\textbf{Shila’s Algorithm [SHI08]}

Shila has proposed an algorithm to protect against security attacks specifically on the ETX protocol. The algorithm uses the ETX metric value to find a detection threshold ($d_{thresh}$). $d_{thresh}$ specifies the minimum number of data packets that should be delivered along a given route. The detection threshold $d_{thresh}$ of a route is computed as the inverse of the summation of the ETX values of all the links from the source node, $S$, to the destination node, $D$. 
\[ d_{\text{thresh}} = \frac{1}{ETX_{S\rightarrow D}} \]  

Equation 3.5

This \( d_{\text{thresh}} \) value will be used to find an *Acceptance Rate* (AR) which will then be used to identify whether or not there is a malicious node en-route. AR is calculated by Equation 3.6, i.e.

\[ AR = n_{\text{transmitted}} \times d_{\text{thresh}} \]  

Equation 3.6

where, \( n_{\text{transmitted}} \) is the number of packets transmitted by the source node.

During data transmission, the source node will randomly include a *Control* packet into the data transmission stream. This *Control* packet contains \( n_{\text{transmitted}} \). When the destination node receives the control packet, it checks whether \( n_{\text{transmitted}} \) is equal or more than \( AR \). If yes, the destination node replies to the source node with a *Positive Control ACK*. Otherwise, a *Negative Control ACK* will be sent. If the source node receives a *Negative Control ACK* or did not receive any ACK, the source node will conclude that there is a malicious node en-route.

However, as mentioned earlier, in a MANET, there are other factors, such as nodes’ mobility and battery black out, etc, which may cause packet loss. If we simply assume poor performing routes contain malicious nodes and exclude them from routing selections, we may end up with too few routes or no routes for communication. In other words, this algorithm takes a very pessimistic view on packet loss; it regards lost packets are always caused by malicious nodes, and blacklists them and excludes them from routing selections. In the worst-case scenario, a network may be partitioned due to lack of available routes. This solution, therefore, may not be the most appropriate one to counter black hole attacks in a network where network topologies change dynamically.

### 3.5.2 Comparing SETX against Shila’s Algorithm

As mentioned above, the Shila’s algorithm [SHI08] relies on the number of data packet the destination node receives to judge if there is a black hole node en-route. The algorithm assumes that if a destination node receives a packet with a rate lower than the *Acceptance Rate* (AR), there is a malicious node included in the route.
As we know, the data-forwarding phase begins after a route has been chosen. This means, with Shila’s algorithm, a black hole node cannot be detected before data forwarding phase. In other words, a black hole node may falsify the forward delivery ratio during the route discovery phase, and once being selected en-route, the black hole node starts dropping data packets. Then the black hole node can be detected by using the Shila’s algorithm.

In some cases, a malicious node may falsify routing information with the intention of being included en-route, and the purpose is to intercept/eavesdrop the traffic. It does not plan to drop any packets. In this way, the Shila’s algorithm will not be able to detect the malicious node.

With the SETX protocol, on the other hand, the detection of any black hole node is carried out during the route discovery process. If a node falsifies the forward delivery ratio, it risks of being detected, and not be included en-route before the data transmission phase starts. Of course, as discussed earlier, even with the SETX protocol, there is still some chance for a black hole node to be selected en-route. This is due to either the black hole node really obtains the best route, or it uses more sophisticated attacks like cooperative black hole attacks. In this case, the SETX protocol cannot detect the black hole attacks.

However, since both solutions are operating in different phases, it is possible to combine the SETX protocol with the Shila’s algorithm to detect black hole attacks. The SETX protocol can be used during the route discovery phase. Once a route has been selected, Shila’s algorithm can be used to monitor a malicious node on dropping attacks. Integrating both solutions will most likely provide a higher level of protection against black hole attacks in the network.
3.5.3 Summary

Table 3.3 shows the comparison with other solutions against black hole attacks. The table includes each solution with its weakness and strength.

<table>
<thead>
<tr>
<th>Solution Name</th>
<th>Protocols</th>
<th>Security Attacks</th>
<th>Dropping Attacks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deng's solution [DEN02]</td>
<td>AODV</td>
<td>Detected by verifying a reported route</td>
<td>Vulnerable</td>
</tr>
<tr>
<td>Tamilselvan’s solution [TAM07]</td>
<td>AODV</td>
<td>Detected by estimating time delay for a round trip of a packet</td>
<td>Vulnerable</td>
</tr>
<tr>
<td>Djenouri’s solution [DJE08]</td>
<td>AODV</td>
<td>Detected by listening to neighbouring nodes and using a two hop acknowledge</td>
<td>Vulnerable</td>
</tr>
<tr>
<td>Shila’s Algorithm [SHI08]</td>
<td>ETX</td>
<td>Vulnerable</td>
<td>Detected by comparing an estimated number of data packet received with the actual data packet received</td>
</tr>
<tr>
<td>SETX</td>
<td>ETX</td>
<td>Detected by confirming the number of probe message sent and received</td>
<td>Vulnerable</td>
</tr>
</tbody>
</table>

Table 3.3: The Comparison of Different Security Solutions against Black Hole Attacks

3.6 SETX Protocol Performance Evaluation

3.6.1 Simulation Modelling

3.6.1.1 Routing Models

The SETX routing model is based on the AODV-UU [AODVUU]. AODV-UU is a Linux implementation of the AODV [RFC3561] routing protocol, developed at Uppsala University, Sweden. It runs as a user-space daemon, maintaining the kernel routing table. AODV-UU was written in C language and it has been released under the GNU General Public License (GPL). AODV-UU implements all mandatory and most optional features of AODV. Hence, AODV-UU is practical and usable in a machine running on Linux. The
development of the AODV-UU has been further ported to a network simulator called NS2 (Network Simulator 2) [NS2].

The SETX routing model extends the functionality of the ETX routing model based on this AODV-UU implementation. There are two major modifications on the AODV-UU implementation to support SETX. The first modification is that a neighbouring node is not allowed to calculate a forward delivery ratio value and advertise it to an initiator node using a hello packet. Instead it is required to store the received probes in its buffer and send these probes to the initiator using RREP packets when necessary.

The second modification is a node attaches corresponding acknowledge probes to the outgoing RREP packets before sending the packets to its upstream node. This means that the size of a RREP packet has to be resized to accommodate the probes. By using this method, SETX values are not updated when each hello packet is received, but rather they are updated on-demand when RREP packets are received.

3.6.1.2 Mobile Node Models

There are four types of mobile nodes defined in the network: 1) source node; 2) destination node; 3) intermediate node; and 4) malicious node.

(1) The source node generates traffic and sends it to the destination node. The traffic is generated when the route is established.

(2) The destination node, often referred to as the sink node, receives the traffic sent by the source node. Performance metrics (e.g. throughput and packet delivery ratio) are measured at this node.

(3) One or more intermediate nodes are connected to each other to form a route between the source node and the destination node. This route can be used to forward traffic from the source to the destination node. The source and destination nodes are not considered as an intermediate node. The location and the waypoint of these nodes are random, using the random waypoint model [BOU05].

(4) A malicious node forges an ETX value. It lures traffic which is to be forwarded through it. Once it receives a data packet, it drops the packet to perform the black hole attack in order to interrupt the network operation. The malicious node also
uses the random waypoint model for its movement. More details of the malicious node model are explained below.

### 3.6.1.3 Malicious Node Model

A malicious node model defines the behaviour of malicious nodes in a network. In this simulation, we use a black hole malicious node model. A malicious node in this model behaves as a black hole node. It forges a routing metric value (e.g. a false forward delivery ratio for the ETX metric) to lure traffic being forwarded through it. It then drop the data packets to interrupt the network operation. The behaviour of the black hole node has been discussed in Section 3.3.8. The TCL command implemented in this model is shown in Appendix B.1.

### 3.6.2 Simulation Parameters

Fine-tuning simulation parameter values may make a difference in investigating how well a routing protocol performs. To evaluate and compare the SETX and ETX protocols, this section discusses simulation modelling and investigates optimum parameter settings for the simulation of both protocols. It defines statistics that will be used to evaluate the performance of the protocols. These statistics are used to measure the performance of the protocols under various parameter settings. Based on the simulation results, recommendations will be given on the selection of the parameter values.

**Mobile Node**

Mobile nodes have a maximum radio range of 50m (See Appendix B.2 for the details of configuration). The wireless standard used is 802.11g [IEEE03], which has the maximum signalling data rate of 54Mbps (Mega bit per second). Please note that at the range of 50m (the maximum range), the signalling data rate will be lower than the maximum signalling data rate. We are aware that there are newer standards of IEEE 802.11 (e.g. 802.11n). These versions have a much higher maximum data rate than the 802.11g. However, the NS2 version 2.26 which is used in the simulation does not support these newer standards natively. It needs a patch to integrate such standard into this particular version. At the time of writing, there is a patch for 802.11n from National of Taiwan University [NTU]. However, it is not compatible with NS2 version 2.26. Therefore, the newest possible
version that we can use is 802.11g. With the IEEE 802.11g, the maximum bandwidth of this standard is 54Mbps. This bandwidth will be used for all of the simulations in this chapter.

**Mobility Class**

In the simulation, a node may move with different speeds. Nodes may have one of two different speeds depending on the scenarios. These are 0m/s (node is not moving) and 1.4m/s (node is moving at a walking speed [BRO06]). Scenario 1, 2, 4 and 5 are simulated with no movement of intermediate nodes. However, Scenario 3 is simulated with the movement of nodes.

**Traffic Model**

Sensor networks normally generate monitoring data which has the same data size. Constant Bit Rate (CBR) traffic will be used in this simulation as it represents typical data that retrieved from sensor networks. The traffic is sent from the source node to the destination node with different data rates. Data rates used in the simulation are 1, 10, 100 and 1,000 packets per second. The different data rates are used in the simulation to investigate if the SETX protocol will perform differently to the ETX protocol.

**Packet Size**

The use of different data packet sizes may affect network performance [SHA12]. A bigger data packet size may not always provide the best throughput rate than a smaller data packet size. Here, different data packet sizes are used in the simulation to see if they affect the performance of our SETX protocol and the ETX protocol. The data packet sizes used in our simulation studies are 800, 1000, 1200, and 1400 bytes.

**Buffer Queue Length**

According to [POR12], the default buffer queue length of 50 packets in NS2 may affect the performance when the data rate is high. The optimum buffer queue length used with a high data rate is 1,000 packets. Therefore, in our simulation a queue length is set at 1,000 packets as recommended.
Energy Model

Regarding to the energy model, we use the power consumptions recommended in [CHE01] which has been conducted from a real network interface card (i.e. Cabletron Roamabout 802.11 DS High Rate). The power consumptions of this card are 1,400mW for a transmission, 1,000mW for a reception, 830mW when idle and 130mW when the card is in a sleep mode. In addition, each node in the simulation will have an initial energy of 100 Joules.

Simulation Setup Summary

In our simulation study reported in this thesis, NS2 version 2.26 is used. The locations of all mobile nodes are limited to a 300m x 100m network area. The nominal radio range of each mobile node is 50m. The traffic is generated using CBR traffic at the source node with different data rates: 1, 10 and 100 packets per second in order to allow nodes to have some buffer before drops the packet. The sizes of data packets are fixed at 800, 1000, 1200 and 1400, respectively. The simulation will run until the there is no data packet delivered (due to no route between the source and the destination node). Table 3.4 summarises these parameters settings.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>NS2 Version</td>
<td>2.26</td>
</tr>
<tr>
<td>Node Movement Speeds</td>
<td>0, 1.4m/s</td>
</tr>
<tr>
<td>WLAN Standard</td>
<td>802.11g</td>
</tr>
<tr>
<td>Maximum Bandwidth</td>
<td>54Mbps</td>
</tr>
<tr>
<td>Nominal Radio Range</td>
<td>50m</td>
</tr>
<tr>
<td>Signal Strength Reception Threshold</td>
<td>7.69113e-08</td>
</tr>
<tr>
<td>Carrier Sensing Threshold</td>
<td>7.69113e-08</td>
</tr>
<tr>
<td>Data Rate (packets/second)</td>
<td>1, 10, 100 and 1,000</td>
</tr>
<tr>
<td>Buffer Queue Length</td>
<td>1,000 packets</td>
</tr>
<tr>
<td>Traffic Type</td>
<td>UDP</td>
</tr>
<tr>
<td>Data Packet Type</td>
<td>CBR</td>
</tr>
<tr>
<td>Application Data Payload Size</td>
<td>800, 1000, 1200 and 1400 bytes/packet</td>
</tr>
</tbody>
</table>
### Simulation Results

Different simulation scenarios are conducted to evaluate the performance of SETX. The results of these simulation scenarios are discussed in terms of Average throughput, packet delivery ratio, control packet count, control packet rate, and simulation duration.

Average throughput is the average rate of successful data delivery measured at the destination node (bytes) divided by simulation duration (seconds) \([\text{KETT}]\). Simulation duration measures the time elapsed from the time that the source node starts sending a RREQ to the time the destination node receives the last data packet.

\[
\text{Average Throughput} = \frac{\text{Total data received at the destination node (Kbytes)}}{\text{Simulation Duration (Seconds)}} \tag{3.7}
\]

Packet delivery ratio is the ratio of the number of data packets actually received at the destination node and the number of data packets sent by the source node and multiplied by 100. This metric measures how well the routing protocol performs in a malicious environment. The higher the number of dropped data packets, the lower the packet delivery ratio is.

\[
\text{Packet Delivery Ratio} = 100 \times \frac{\text{Number of Delivered Packets}}{\text{Number of Dispatched Packets}} \tag{3.8}
\]

Control packet count refers to the number of control packet transmissions. If a RREP packet is sent over a 5 hop route to the source node, this will be counted as 5 packets using

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Mobile Nodes</td>
<td>2, 6, 11</td>
</tr>
<tr>
<td>Initial Energy</td>
<td>100 Joules</td>
</tr>
<tr>
<td>Energy Consumption in Transmitting Packets</td>
<td>1400 mW</td>
</tr>
<tr>
<td>Energy Consumption in Receiving Packets</td>
<td>1000 mW</td>
</tr>
<tr>
<td>Energy Consumption in IDLE State</td>
<td>830 mW</td>
</tr>
<tr>
<td>Energy Consumption in Sleep State</td>
<td>130 mW</td>
</tr>
</tbody>
</table>

Table 3.4: Simulation Parameter Settings
this metric. Control packet rate is slightly different, it is calculated from control packet count divided by the simulation time.

\[
Control \ Packet \ Rate = \frac{\text{Control Packet Count (packets)}}{\text{Simulation Duration (seconds)}} \quad \text{Equation 3.9}
\]

We investigate the overhead incurred in operating the SETX protocol, compared with that of the ETX protocol in Scenarios 3.1-3.3. This investigation is done assuming a network environment is without any malicious behaviour. We then investigate the performance of the SETX protocol when there are malicious behaviours in the network and compare it to that of the ETX protocol using Scenarios 3.4 and 3.5. Each scenario has been set with different data rates (i.e. 1, 10, 100 packets per second), and different data payload sizes (800 bytes, 1000 bytes, 1200 bytes and 1400 bytes). The details and the results of these simulations are reported below.

3.6.3.1 Scenario 3.1: No Intermediate Node

This scenario is designed to compare the performances of the SETX and ETX protocols when there is no intermediate node involved. The source node is connected directly to the destination node. The locations of the two nodes are fixed as shown in Figure 3.41.

![Figure 3.41: Network Topology of Scenario 3.1](image)

The circle stands for a mobile node. The "S" letter in the circle means this node is the source node. The "D" letter is for the destination node. The arrow means there is a wireless connection between two nodes. Data traffic is flowing following the arrow direction. The letter below a node indicates the location of the node in (x, y) format.

With this scenario, there is no intermediate node involved. There is also no movement of a node. We are able to see the real differences of protocol overheads between SETX and ETX with this simple topology.
The full simulation results for this scenario are shown in Appendix D.1. Here Figure 3.42 shows the data rate of 1 packet per 1 second. We can see that the average throughput is slightly lower than the size of a data packet. In fact, it is 20 bytes lower than the size of the data packet. This is because each data packet also contains an IP header. As this simulation is done based on IPv4 and the standard IP header is 20 bytes, the average throughput results shown in the figure are valid.

![Average Throughputs in Scenario 3.1 (1 packet/second)](image)

Figure 3.42: Average Throughputs in Scenario 3.1 with the data rate of 1 packet/second

When the data rate increases, the average throughput also increases. This is because the source node sends more packets within the same time period. For example, instead of sending 1 packet per second, we increase the data rate to 10 packets per second. In this case, the source node will send 10 times more packets than the previous one. Obviously, the destination node will receive more packets provided the network is not congested. As a result, the average throughput is higher according to the data rate.
Figure 3.43: Average Throughputs in Scenario 3.1 with the data rate of 10 packets/second

Figure 3.43 shows the average throughputs when the data rate is set to 10 packets per second. The average throughputs are 10 times higher than the results when the data rate is 1 packet per second, which is what we expect. Figure 3.44 shows the average throughput results when the data rate is increased to 100 packets per second. When comparing ETX and SETX in this scenario, the average throughputs of both protocols are virtually identical. This result is within our expectation.

The simulation results presented in Figure 3.45 show that the simulation duration of the SETX protocol is shorter than the ETX protocol. This is because when the size of a data packet is bigger, more energy is required to transmit the packet. In other words, the longer
the packet, the more energy a node requires to transmit/or receive it, thus the shorter the simulation duration. As the packet size used in the SETX protocol is longer than that used in the ETX protocol. So the simulation duration for the SETX protocol is shorter.

Figure 3.45: Simulation Durations in Scenario 3.1 with the data rate of 1 packet/second

Figure 3.46: Simulation Durations in Scenario 3.1 with the data rate of 100 packets/second

When the data rate goes higher, the simulation duration goes shorter. Figure 3.46 shows that the simulation duration is reduced markedly when the data rate is set to 100 packets/second, in comparison with the case when the data rate is set to 1 packet/second (from Figure 3.45). This is because each packet transmission consumes battery. A higher data rate means more packet transmissions, thus more battery consumption. As a result, the simulation durations are shorter.
Figure 3.47 shows the number of control packets generated by ETX and SETX protocols. From the figure, it can be seen that ETX generates more control packets than SETX. The reason behind this is that the simulation duration for the ETX protocol is longer than that for the SETX protocol. When the simulation duration is longer, nodes will generate more traffic, including control traffic.

![Control Packet Counts in Scenario 3.1 (1 packet/second)](image1)

Figure 3.47: Control Packet Counts in Scenario 3.1 with the data rate of 1 packet/second

![Control Packet Counts in Scenario 3.1 (10 packets/second)](image2)

Figure 3.48: Control Packet Counts in Scenario 3.1 with the data rate of 10 packets/second

When the data rate is higher, the simulation duration will be shorter. So there were less control packet count. The results in Figure 3.47 and Figure 3.48 show that the control packet count for the data rate of 10 packets/second was significantly lower. This was not expected. However, further investigation has revealed that this is due to the simulation
duration for the data rate of 10 packets/second is shorter than the simulation duration for
the data rate of 1 packet/second. Since the simulation duration is shorter, nodes have less
time to generate control packets. As a result, the control packet count for the data rate of
10 packets/second is smaller than the control packet count for the data rate of 1
packet/second.

The control packet rates for the two cases are very similar. The control packet rates for the
data rates of 1 and 10 packets/second are shown in Figure 3.49 and 3.50, respectively.
Both figures show that SETX and ETX have very similar control packet rates which are 2
control packets/second in this scenario.

Figure 3.49: Control Packet Rates in Scenario 3.1 with the data rate of 1 packet/second

Figure 3.50: Control Packet Rates in Scenario 3.1 with the data rate of 10 packets/second
The packet delivery ratios of this scenario was also as expected. The destination node received all the data packets sent from the source node. With all of the data rates, and all of the packet sizes we simulated, the packet delivery ratios are all 100%.

Figure 3.51: Packet Delivery Ratios in Scenario 3.1 with the data rate of 1 packet/second

### 3.6.3.2 Scenario 3.2: With Intermediate Nodes

Scenario 3.2 compares the performances between ETX and SETX when there are intermediate nodes involved. The network topology used for this scenario is shown in Figure 3.52.

![Network Topology of Scenario 3.2](image)

Comparing the average throughputs in Scenario 3.1 and Scenario 3.2, it can be seen that the average throughputs of both scenarios are virtually the same when the data rate is low. This is because there is ample bandwidth to forward the traffic, and they are not so busy to forward data packets along the route.
The control packet rates of this scenario are higher than that in Scenario 3.1. This is due to the fact that there are intermediate nodes taking part in the packet transmissions. When these intermediate node request or update a route with their neighbouring nodes, the routing packets are also counted. The more number of nodes involved in packet forwarding, then the higher control packet rate is. The control packet rates of this scenario with the data rate of 1 packet/second is shown in Figure 3.54. The full simulation results for this scenario are shown in Appendix D.2.
3.6.3.3 Scenario 3.3: With Moving Intermediate Nodes

This scenario investigates whether the movement of intermediate nodes would affect the performance of SETX when compared with ETX. The scenario shows 4 intermediate nodes moving vertically up and down. The intermediate nodes 1 and 3 start from the locations (15, 85) and (45, 85), and then move down to (30, 15) and (60, 15) with the speed of 1.4m/s. Once they reach their destinations, they move up to their starting locations, and then move back down again. These movements continue until the simulation ends. Intermediate nodes 2 and 4 also do the same movements, but they start at the bottom locations instead of the top locations. This scenario is set to see whether moving intermediate node will affect the performance between SETX and ETX protocols. The network topology of this scenario is illustrated in Figure 3.55.

Figure 3.55: Network Topology of Scenario 3.3

The full simulation results of this Scenario 3.3 are shown in Appendix D.3. Here, we compare the simulation results of two data rates, 100 packets/second and 1,000 packets/second. The throughput results of these simulations are given in Figure 3.56 and Figure 3.57, respectively. From these figures, it can be seen that as the packet size increases, the throughput values also increase. This is because a longer packet size can reduce the overhead proportion, and if the network is not heavily loaded, this can lead to an increase in throughput.
Figure 3.56: Average Throughputs in Scenario 3.3 with the data rate of 100 packets/second

Figure 3.57: Average Throughputs in Scenario 3.3 with the data rate of 1,000 packets/second
We have also compared the result shown in Figure 3.57 to those results collected from Scenario 3.2 (shown in Figure 3.58). The results here are very interesting. The average throughputs in Scenario 3.3 is significantly higher than those in Scenario 3.2. Here, we searched for the reason behind the higher average throughput results shown in Figure 3.57 in comparison with those shown in Figure 3.58. It turned out that the packet delivery ratios (shown in Figure 3.59) of the Scenario 3.3 are higher than those in Scenario 3.2 (shown in Figure 3.60).
Figure 3.60: Packet Delivery Ratios in Scenario 3.2 with the data rate of 1,000 packets/second

After searching for the reason for a long time, we have discovered that the network topology could be the reason to make the delivery ratio in Section 3.3 higher than the Scenario 3.2. In Scenario 3.3 the routes selected by the source node were S→1→3→D and S→2→4→D. These routes were shorter than the one in Scenario 3.2. Since the number of hop counts is shorter, a chance that two packets will be transmitted at the same time is lower in Scenario 3.3. Therefore, the chance that packets will be lost is lower too. For this reason, we believe that this has made the packet delivery ratios in Scenario 3.2 is lower than those of the Scenario 3.3.

### 3.6.3.4 Scenario 3.4: With a Malicious Node outside the Best Route

This simulation investigates the performances of the SETX and ETX protocols in the presence of black hole attacks. As shown in Figure 3.61, this scenario defines two routes to the destination node. The first route (i.e. S→1→2→3→4→5→D) has 6 hops from the source node to the destination node while the second route (i.e. S→6→7→8→9→D) has only 5 hops. The first route has a malicious node en-route while the second route does not. The black hole node (shown as node 5) fabricates the ETX value in order to be included en-route. Once the traffic is sent through the black hole node, it will drop half of the traffic.
Here, the second route is assumed to be the best route according to the ETX metric. Assuming that each of the links has an ETX value of 1.0, as the first route has 6 links, the first route has the ETX value of 6.0. Similarly, the ETX value of the second route is 5.0. As the second route has a lower ETX value (which is more preferable), the second route is therefore selected.

The simulation results of this scenario are as expected. That is, the ETX protocol selects the first route (i.e. S→1→2→3→4→5→D) whilst the SETX protocol selects the second route. This is because the black hole node (in the first route) fabricates the ETX value and tells the source node that it has the best route to the destination node. The source node then believes the black hole node and selects the first route instead of the second route. On the other hand, with the SETX protocol it is much harder for the black node to fabricate the SETX value. Because the metric value is not fabricated, the source node selects the second route as it is the best route.

When the ETX protocol is used, the average throughput and packet delivery ratio results are halved in comparison of the results from the SETX protocol. This is because the black hole node drops half of the traffic, and forward the other half to the destination node. See Appendix D.4 for the full simulation results for this scenario. The average throughput and the packet delivery ratio results of the data rate is set to 100 packets/second are shown in Figure 3.62 and Figure 3.63, respectively.
With the SETX protocol, however, the black hole node cannot easily manipulate the routing decision made by the source node. So the source node would be able to still select a route with the lowest SETX value (i.e. the best route). Since this route does not include the black hole node, the average throughput and the packet delivery ratio results are much higher (i.e. near 100% packet delivery ratio).

One question may be asked with ETX protocol why the source node does not change to the second route when a significant number of packet are dropped by an intermediate node. This is because the ETX protocol does not have a built-in mechanism to respond to packet loss due to black hole attacks. The neighbouring nodes of the black hole node still have a
route to the destination node registered in their routing tables. The wireless links still exist. Route repair or discovery processes are not activated in such cases. Therefore, the source node still uses the first route.

Another question is how we can handle this problem. There are two possible approaches: (1) do not select a route containing a black hole node in the first place, and (2) change for a better route when a packet delivery ratio drops below a threshold value (e.g. 50%). The SETX protocol is designed to support the first approach by developing a mechanism to allow a source node acquiring a more accurate ETX metric value. This is so that the source node stands a better chance to find the best route, and have less chance to be affected by a black hole node en-route. The second approach may be done by applying a trust based method. So at the beginning of the transmission process, the source node may select a route with a black hole node, but after a while the packet delivery ratio will be decreased. When the destination node detects that the packet delivery ratio is lower than a defined threshold value, the destination node may initiate a route repair. Then an alternative route can be searched. However, when searching for another routing this time, a trust metric related type (i.e. Section 2.3.3.4) or the packet delivery ratio history of a route will need to be taken into consideration. Otherwise, the same route (i.e. the route with the black hole node) will be selected, because the black hole node will fabricate the metric value again.

The SETX protocol is designed to deal with black hole attacks in the situation where the location of black hole nodes is not in the best route. The protocol cannot avoid selecting a route where the black hole node is included in the best route. This situation is to be simulated in Scenario 3.5. In this scenario, we shall see how SETX and ETX protocols react when a black hole node is included in the best route.

### 3.6.3.5 Scenario 3.5: With a Malicious Node inside the Best Route

This scenario uses the same network topology as the previous scenario (Scenario 3.4) except that the location of the black hole node is different. The black hole node in this scenario is intermediate node 9. It is included in the second route. Whether or not the black hole node fabricates the ETX value, both ETX and SETX would select the second route as it is the best route.
The full simulation results for this Scenario are shown in Appendix D.5. show that for both the SETX and ETX protocols the packet delivery ratios (shown in Figure 3.65) are about 50%. This is because the black hole node has dropped half of the data packets routed through the route. The average throughputs from both protocols, as shown in Figure 3.66, are virtually the same. They are about the half of the throughput under the condition where there is no black hole node attack.

Figure 3.65: Packet Delivery Ratios in Scenario 3.5 with the data rate of 100 packets/second
Most of the simulation results shown that the performance of SETX and ETX are very close when there is no black hole node in the network (e.g. Scenario 3.1, 3.2 and 3.3), or even when there is a black hole node located in the best route (e.g. Scenario 3.5). However, when there is a black hole node located outside the best route (e.g. Scenario 3.4), the performance of the SETX protocol is much better than the performance of the ETX protocol, as with the ETX protocol the black hole node would be able to forge routing metric values and lure the traffic being routed via the black hole node before performing packet dropping attacks. This is the advantage of using the SETX over the ETX protocol.

### 3.7 SETX Limitations

The aim of SETX protocol is to provide a method to prevent black hole nodes from advertising a fabricated forward delivery ratio of a wireless link between itself and one of its neighbours. The SETX protocol allows a node to select a route which is more trustful than that selected by the ETX protocol, especially when there is a black hole node in the network. With the probe verification mechanism of the SETX, it is difficult for a black hole node to fabricate a metric value. However, the SETX protocol cannot prevent a source node from selecting the best route that includes a black hole node en-route.

A trust management scheme can be used to address this limitation. A trust management scheme [PIR06] can be used for nodes to monitor the behaviour of their neighbours. If
their neighbours have intentionally dropped packets, their trust levels will be affected. If the trust level of a neighbouring node drops below a given threshold, this neighbouring node would be considered as a malicious node, and should be excluded in a routing operation. This trust based approach to countering black hole attacks means that a black hole node may be selected en-route and be able to attack the network (i.e. drop the packets) for a while before it is detected. Once it is detected, an alarm can be sent out to inform other nodes, so that it is excluded in routing operations.

This issue is addressed by designing the FRD framework that supports the use of multiple routing metric types in routing decision making. The framework can consider both SETX and trust metric type at the same time. In addition to that, the framework also considers the requirements of different application-layer data types to make the best routing decision for the application.

### 3.8 Chapter Summary

This chapter has presented the design and simulation study of a novel solution to counter black hole attacks on the ETX protocol. This novel solution, i.e. the SETX protocol, shifts the task of computing link quality metric values from the receiving end of a communication onto the initiator of a communication. It has a built-in mechanism to allow the initiator to verify any probe messages (used to estimate the link quality) returned by the neighbouring nodes, thus effectively reducing the chance of successfully fabricating link quality data by malicious nodes. Simulation results have shown that the SETX protocol provides a much better performance than its ETX counterpart in malicious environments. The larger the network size, the bigger the improvement, and these improvements are achieved with very little overhead costs.
Chapter 4

A Flexible Routing Decision (FRD) framework for MANETs

4.1 Chapter Introduction

This chapter introduces a novel method called the Flexible Routing Decision (FRD) framework. This framework is designed for routing decision making in MANETs. It uses a cross-layer approach to support application-level QoS requirements by allowing users to use and rank different types of routing metrics. It supports the use of different Multi-Criteria Decision Making (MCDM) techniques in routing decision making. Furthermore, the chapter critically analyses the FRD framework and evaluates its performance using the NS2 simulation package. The novelty of the FRD framework lies in its ability to select routes for an individual application data type based on its QoS and security requirements in a single platform.

In detail, Section 4.2 describes the problem to be tackled by the FRD framework. Section 4.3 critically analyses existing routing algorithms in MANETs, identifying their pros and cons. Section 4.4 discusses the background of Multi-Criteria Decision Making techniques and how to put them into MANET routing decision making context. Section 4.5 presents the novel FRD framework. Section 4.6 evaluates the framework using NS2. Finally, Section 4.7 concludes the chapter.

4.2 Problem Statement

Wireless devices are increasingly capable. Applications that used to be only accessible on desktop computers are increasingly runnable on such devices. Users may run several different applications concurrently on the same device. Different applications may generate different data types, which, in turn, may impose different QoS and security requirements. These requirements may better be satisfied by forwarding the data along different routes. This leads to the need for selecting different routing criteria (i.e. routing
metric types) and selecting routes based on these criteria, for a given set of QoS requirements. Take a military ad hoc network communication scenario as an example [MAN09]. Soldiers are equipped with mobile devices. They use these devices to communicate with each other. They may use an email application to convey information on their physical conditions, locations or other battlefield information to their commanders. In the meantime, they may also talk to each other using the Voice over IP (VoIP) technology. In this example, each user is running two applications simultaneously; one is email and the other is voice communication. As the email messages are highly sensitive and urgent, a route with a sufficiently high security and reliability level should be chosen for the data generated from the email application. With regard to data generated from the voice communication, in addition to confidentiality requirement, which can be addressed by using an end-to-end encryption service, a lower level of delay and jitter is also important. Thus, the route with the shortest delay may be the most appropriate.

Existing ad hoc network routing solutions do not consider these application-level requirements when making a routing decision. They typically make a routing decision only based upon the information acquired at the network layer. For example, the Trust-based Routing Protocol (TRP) [XUE04] uses a trust value as the underlying routing criterion to select the most trustworthy and reliable route among those available. A route selected using this method may be appropriate for forwarding reliability-sensitive data such as financial transactions, but may not be the best route for data generated by a delay-sensitive application. This is because the most trustworthy route may not be the shortest route or one with the least delay. This means that using a single metric type and/or information acquired at the network layer alone may not be sufficient to accommodate different QoS and security requirements imposed by diversified user-level applications or application-level data types.

Here, we hypothesise that making a routing solution with a consideration of application-layer requirements could satisfy the needs of the application in terms of reliability, security, and performance better than without considering one. To validate this hypothesis, we have developed a novel framework, the Flexible Routing Decision (FRD) framework, which uses a cross-layer approach to map application layer QoS parameter values to routing metric values used by the network layer.
Generally, a routing decision is made by a routing algorithm. A routing algorithm is a mechanism used to find the best route from available routing candidates. Each routing algorithm has a different mechanism to find its best route. For example, a distance vector routing algorithm selects a route based on a hop count. On the other hand, the routing algorithm of the ETX routing protocol finds a route by using a delivery ratio of probes (as mentioned in Section 2.3.3.3). The FRD framework makes use of both application-layer requirements and network layer information to find a route. Before introducing the routing algorithm of the FRD framework in detail, the following section first gives a review of existing routing algorithms used in MANET routing protocols.

4.3 A Literature Review: Routing Algorithms for Routing Decision Making

MANET Routing algorithms can be divided into two categories. The first category of algorithms uses a single routing metric type to select routes. Hereafter, these algorithms will be referred to as single metric type based routing algorithms. The second category will be referred to as multiple routing metric types based routing algorithms. The latter algorithms select routes using two or more routing metric types.

4.3.1 Single Routing Metric Type Based Routing Algorithms

As the name suggests, a single routing metric type based routing algorithm only uses one routing metric type to evaluate and select routes. A routing decision making process of this kind of algorithms is straightforward. A routing candidate with the best routing metric value will be selected. For example, the distance vector algorithm (the most common algorithm in MANET routing [PER94, PER97, JOH96]) uses a hop count as its routing metric type. It selects a routing candidate with the lowest hop count value (i.e. the shortest route) to the destination node. The algorithm makes the decision based on a packet traversing the shortest route with the potential to experience the least delay. That is why the shortest routing candidate is considered as the best route for this routing algorithm. However, if the shortest route is always chosen, traffic can build up along this route, after which traffic may experience a longer delay. In addition, the intermediate nodes along the
route could be overloaded by the flood of traffic. As mobile nodes are operated by battery power, continuously using the same intermediate node can quickly exhaust the node’s battery power and cut the node off the network. As a result, the network may be re-partitioned. To address these problems, the second category of ad hoc routing protocols, i.e. multiple routing metric type based routing algorithms, are proposed.

4.3.2 Multiple Routing Metric Type Based Routing Algorithms

In a multiple routing metric type based routing algorithm, a routing decision is made by considering two or more routing metric types. These routing protocols [TSI01, XUE03, ZHU02] uses a multi-metric based routing algorithm to evaluate and compare metric values of two or more routing metric types of all routing candidates, then select the best route. When several routing metric types are used, the routing algorithm typically uses one of the following techniques to select the best route: Rank Order and Threshold methods.

4.3.2.1 The Rank Order Method

The rank order method is one of the most commonly used decision making methods in multi-metric based routing algorithms. For example, it is used in the design of a routing protocol called OSLR [RFC3626]. In this protocol, the rank order method is used to rank multiple routing metrics based upon their priorities. The metric with the highest priority (called the first rank metric) will be considered first. If there are several routing candidates with the same highest first rank metric value, all of these routing candidates will be maintained in the routing candidate list, while other routing candidates, i.e. those with lower first rank metric values, will be discarded. The routing candidates maintained in the list will then be further sorted based on the second rank metric values – with the higher value first. When making the final routing decision amongst the candidates with the highest first rank metric value, the one with the highest second rank metric value will be selected. If there were two or more routing candidates with the same highest second rank metric value, then the third rank metric values will be used to rank and select a candidate, and so on.
Let us use an example to further illustrate this method. Say, a source node, \( S \), has a routing table as shown in Table 4.1. The table uses three metric types, Metric Type A, Metric Type B and Metric Type C. Metric Type A is the first rank metric type (i.e. having the highest priority). Metric Type B is the second rank and Metric Type C is the third rank metric type. There are three routing candidates available for \( S \) to choose from, i.e. Route A, Route B and Route C. Route A has the lowest value of the first rank metric type (its Metric Type A has the value of 0.5). Route B and Route C both have the value of the first rank metric type of 0.6. So Route A will be discarded first. In the next stage, as routes B and C have the same value of the first rank metric type, they will be further compared by using the values of the second rank metric type (i.e. Metric Type B). We can see that Route B has a higher value of the second metric type, so Route B will be selected by using this routing algorithm.

<table>
<thead>
<tr>
<th>No.</th>
<th>Routing Candidates</th>
<th>Metric Types (Rank)</th>
<th>Metric Type A (1st)</th>
<th>Metric Type B (2nd)</th>
<th>Metric Type C (3rd)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Route A</td>
<td>0.5</td>
<td>0.9</td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Route B</td>
<td>0.6</td>
<td>0.4</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Route C</td>
<td>0.6</td>
<td>0.3</td>
<td>0.9</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.1: The Example of a Routing Table that uses of the Rank Order Method in Multi-Criteria Routing Decision

Most of the time, routing decisions are made based on the first rank metric values. Only when two or more routes share the same highest first rank metric value, the next rank metric values will be considered. This process is straightforward and efficient. It does not consume much additional CPU resources. In other words, this method has a comparable performance with the single metric routing decision making method.

However, as the multi-metric based routing decision method puts an overriding priority on a higher rank metric type, and most of the time it ignores the values of lower ranking metric types, this method sometimes may not lead to an optimal routing decision. Revisiting the example shown in Table 4.1, using the rank order method, Route B was chosen because it has the best values of both the first rank metric type (i.e. Metric Type A) and second rank metric type (i.e. Metric Type B). However, in this example, this method has ignored the value of the third rank metric type (i.e. Metric Type C) altogether. The value of the third rank metric type for Route B is 0.4. This value is the lowest among the available routes. If Metric Type C is the remaining battery level metric type, this route will be the first one to break, as this route has the lowest remaining battery level (i.e. 0.4). If the route breaks before the transmission is complete, then the source node \( S \) will have to
initiate another route discovery process. This could lead to more energy and resource consumptions, as the route recovery process will generate and flood route request packets into the network. In other words, this method often leads to sub-optimal routing decisions, particularly when the low rank routing metric values are much lower than higher rank metric values. This indicates that this method is not efficient. It is addressed by using another multi-metric decision making method the threshold method.

### 4.3.2.2 The Threshold (or $\varepsilon$-Constraint) Method

The threshold method optimises one metric type but sets a threshold value for each of the other metric types. The threshold value indicates a minimum acceptable routing metric value. At run-time, the value of the metric type concerned (say Metric Type A) will be compared with the threshold value specified. Any route that satisfies the threshold value will be maintained in the routing table as routing candidates, whereas the routes that do not satisfy the threshold value will be discarded. The routing decision will then be made by using the same way as the rank order method, i.e. choosing one with the best metric value (say Metric Type B) from those routing candidates listed in the table.

Most of the MANET QoS routing solutions support multi-criteria decision making. These protocols largely use this threshold method to select routes [GER02, XUE03, ZHU02]. Assume that the number of supported routing metric types is $n$ (where $n \geq 2$). The number of controlled metric types with assigned threshold values will be $n-1$, and there will be only one metric type without a threshold value assigned.

Table 4.2 shows that Routing Protocol 1 has a threshold value of 0.4 for Metric Type B and Metric Type C. Here Metric Type B and Metric Type C are called controlled metric types while Metric Type A is a considered metric type (with no threshold). If any routing candidate has the Metric Type B or Metric Type C metric value less than 0.4, that routing candidate will be discarded. Then the routing decision will be decided based on the value of Metric Type A only. In other words, if all routing candidates have the values of Metric Type B and Metric Type C higher than the 0.4 threshold value, a routing candidate with the highest value of Metric Type A will be selected.
Table 4.2: The Example of a Table of Routing Protocols against Metric Types that uses the Threshold Method in Multi-Criteria Routing Decision

<table>
<thead>
<tr>
<th>No.</th>
<th>Routing Protocols</th>
<th>Threshold Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Metric Type A</td>
</tr>
<tr>
<td>1</td>
<td>Protocol 1</td>
<td>Considered</td>
</tr>
<tr>
<td>2</td>
<td>Protocol 2</td>
<td>0.4</td>
</tr>
<tr>
<td>3</td>
<td>Protocol 3</td>
<td>0.4</td>
</tr>
</tbody>
</table>

For most of the MANET routing protocols published in the literature (e.g. [KIM02, SIN98, XUE04]), this method is used with only two routing metric types. One of the two routing metric types is usually the hop count metric type, and the other one will be another routing metric type, e.g. a trust or power aware metric type. Typically, the hop count is used as a considered metric type while the other one is assigned with a threshold value.

However, there are further issues that should be addressed regarding this method. The first issue is that sometimes it may be difficult to define a proper threshold value for a given metric type. If a threshold value is set too high, many available routes may be excluded from the routing candidate list, and this may lead to a shortage of routes. On the other hand, if a threshold value is set too low, then unsatisfactory routes may also be considered. This may not be beneficial when making a routing decision in response to QoS requirements. Therefore, selecting a proper threshold value is important and can be difficult in real-life applications.

The second issue is that this method does not consider the real quality of a controlled metric. The actual value of a controlled metric type will not be considered as long as it meets the threshold value. Using Routing Protocol 1 from Table 4.2 and the routing table from Table 4.1 as an example, Route A has the value of Metric Type B of 0.9, while Route B has the value of the Metric Type B of 0.4 and the threshold values of Metric Type B is 0.4. Route C will not be considered because the value of its Metric Type B is lower than the threshold value. Routes A and B will be considered as both of them have the values of Metric Type B and Metric Type C more than or equal to the threshold values. The routing decision will be decided by the metric value of Metric Type A. In this case, the metric value of Metric Type A of Route B is higher than that of Route A, so Route B will be selected. However, if we compare Route B with Route A carefully, Route A has significantly higher value of Metric Type B and Metric Type C. In some case, it might be more efficient to select Route A instead of Route B, since Route A has higher values of both Metric Type A and Metric Type C. This threshold method does not take this scenario
into account. Instead it makes the routing decision based on the value of Metric Type A only, even though the value of Metric Type A of Route B is only slightly better than that of Route A. This observation shows that this method does not always make an optimal decision, especially when the values of controlled metric types are significantly different amongst routing candidates.

In order to address these issues, we need a method that can consider the quality of all routing metric types. On the top of that, if we can also satisfy the application-layer requirements at the same time, the routing decision should be more optimal. Our aim is to use a Multi-Criteria Decision Making (MCDM) technique to design a routing decision method that could address these issues. In light of this, the following section gives a background discussion on MCDM techniques in the MANET routing decision making context.

4.4 Multi-Criteria Decision Making (MCDM) Technique

In the previous section, we have discussed the different routing algorithms in MANETs. Those routing algorithms make a routing decision without considering application-layer requirements. Making a routing decision without taking application-layer requirements into account might not be able to satisfy the requirements well. In this section, we discuss how to make use of a MCDM technique to design a routing decision method that could take both application-layer requirements and multiple routing metric types into account.

MCDM is considered as one of the most well-known techniques of decision making [LAZ09]. It deals with decision situations where the decision maker has several conflicting objectives. In typical real-life situations, no ideal alternative exists in the sense of one that is optimal for all objectives. The task of MCDM is to find a good compromise. This is the technique that performs best in the eyes of the decision maker, taking into account all objectives simultaneously. MCDM can also be applied to a MANET routing decision making by using application-layer requirements and multiple routing metric types.

In MCDM technique, routing metric values used in this method must be linear in order to compare routing metric values between different routing metric types. However, MCDM
technique cannot be used with a non-linear routing metric value. This is because the value of the non-linear routing metric cannot be compared with other routing metric types.

4.4.1 MCDM in MANET Routings

In the remaining part of this chapter, the following notations are used.

- For clarity, we use a normal font to indicate a parameter name, and italic font to indicate a parameter value. For example, RRMT is a name of a routing metric type, whereas $RRMTW$ is a weight value of a metric type.

- A set of $m$ Routing Candidates (RC) is denoted as $RC_1, RC_2, RC_3, ..., RC_m$.

- A set of $n$ RRMTs is denoted as $RRMT_1, RRMT_2, RRMT_3, ..., RRMT_n$.

- $RRMV^{j}_i$ is the metric value for metric $RRMT^j$ of $RC_i$, where $i = 1, 2, 3, ..., m$ and $j = 1, 2, 3, ..., n$.

- $RRMTW^{j}_A$ is the weight of $RRMT^j$ of application data type $A$.

- $WSMS^A_i$ is the Weight Sum Model Score of the route $RC_i$ for application data type $A$. A route with the highest $WSMS$ value is the best route when using WSM technique.

- $WPMR^A(\frac{RC_1}{RC_2})$ is the Weight Product Model Ratio of the route $RC_i$ comparing to $RC_2$ for application data type $A$. A better route is the route that has the $WPMRs$ more than 1 when comparing to another routing candidate.

- A relative $AHP^{j}_i$ value is the actual $RRMV^{j}_i$ divided by the sum of all $RRMVs$ with the same $j$ (i.e. $\sum_{i=1}^{m} (RRMV^{j}_i)$).

- $AHP^A_i$ is the AHP Score of $RC_i$ for application data type $A$.

- A relative $RAHP^{j}_i$ value is the actual $RRMV^{j}_i$ divided by the maximum value of all $RRMVs$ with the same $j$ (i.e. $\max_{1<i<m}(RRMV^{j}_i)$).

- $RAHP^A_i$ is the RAHP Score of $RC_i$ for application data type $A$.

In general, MCDM makes a decision based on given criteria and selects the best from available alternatives. A given criteria in our problem context (i.e. MANET routing) is basically an application-layer requirement (which we transfer into a routing metric type), while alternatives are routing candidates. In MCDM, a given criterion is represented in a form of weight. Here, we can also assign a weight to each Relevant Routing Metric Type.
(RRMT)\(^1\), so that we can decide how important these RRMTs are. The weights are assigned per application. A RRMT with more importance should be assigned with a weight value that is higher than the weight values assigned to less important RRMTs. For example, application data type A has three RRMTs: RRMT\(_1\), RRMT\(_2\) and RRMT\(_3\). For this application, RRMT\(_1\) is considered as the most important metric type, and RRMT\(_2\) is more important than RRMT\(_3\). Given that \(RRMTW^A_1\) is the weight of RRMT\(_1\) for application data type A, the weight values of these RRMTs should be \(RRMTW^A_1 > RRMTW^A_2 > RRMTW^A_3\), and the sum of these values, i.e. \(RRMTW^A_1 + RRMTW^A_2 + RRMTW^A_3\), should be 1.0.

A MCDM technique can be used to make a routing decision. There are four MCDM techniques discussed in this thesis, Weighted Sum Model (WSM), Weighted Product Model (WPM), Analytic Hierarchy Process (AHP) and Revised Analytic Hierarchy Process (RAHP). The following sections discuss how these techniques are used to select the best route.

### 4.4.2 The Weighted Sum Model (WSM) Technique

WSM [FIS67] is the simplest and most commonly used MCDM technique in single dimensional problems, where all the units of RRMT are the same (e.g. seconds, bytes/sec). This approach makes a decision by selecting a route with the highest Weighted Sum Model Score (WSMS). WSMS is calculated from the sum of the performance value of each RRMT. An equation that can be used to calculate WSMS is shown below.

\[
WSMS^A_i = \sum_{j=1}^{n} (RRMV^j_i \times RRMTW^A_j)
\]

Equation 4.1

where \(WSMS^A_i\) is the WSMS of RC\(_i\) for application data type A. \(RRMV^j_i\) is the value of the metric type, RRMT\(_j\), for route RC\(_i\). \(RRMTW^A_j\) is the weight of the RRMT\(_j\) for application data type A. The terms of \((RRMV^j_i \times RRMTW^A_j)\) can be called the performance value of RC\(_i\) in terms of RRMT\(_j\).

---

\(^1\) The term of RRMT means relevant routing metric type. In a network, there may be several routing metric types available. However, some of them may not be relevant to the application data type (thus application-layer requirements) under consideration. RRMT here is used to refer to a routing metric type that is relevant to a given application (an application that we are going to select a route for).
Example Scenario 1

Suppose that application data type A has four RRMTs and three routing candidates. All RRMTs are expressed in exactly the same scale (i.e. in a range between 0 and 100). The relative weights of the four RRMTs assumed to be: $RRMTW^1_A = 0.2$, $RRMTW^2_A = 0.15$, $RRMTW^3_A = 0.4$ and $RRMTW^4_A = 0.24$. The performance values of the RRMTs are summarised in Table 4.3:

<table>
<thead>
<tr>
<th>Routing Candidates</th>
<th>RRMTs ($RRMTW_A$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$RRMTW^1_A(0.20)$</td>
</tr>
<tr>
<td>RC$_1$</td>
<td>25</td>
</tr>
<tr>
<td>RC$_2$</td>
<td>10</td>
</tr>
<tr>
<td>RC$_3$</td>
<td>30</td>
</tr>
</tbody>
</table>

Table 4.3: The Decision Table of Example Scenario 1 when the WSM Technique is used

The WSMS of the three routing candidates are:

$$WSMS^1_A = 25 \times 0.20 + 20 \times 0.15 + 15 \times 0.40 + 30 \times 0.25 = 21.50$$

$$WSMS^2_A = 10 \times 0.20 + 30 \times 0.15 + 20 \times 0.40 + 30 \times 0.25 = 22.00$$

$$WSMS^3_A = 30 \times 0.20 + 5 \times 0.15 + 30 \times 0.40 + 5 \times 0.25 = 20.00$$

Based on these performance values, the best route for application data type A is the route RC$_2$ because it has the highest WSMS of 22.00.

The difficulty with this method emerges when it is applied to multi-dimensional cases where the units/scales of RRMTs are different. Combining different units does not make sense, as it is equivalent to adding two different things, e.g. combining an apple with an orange. Since in MANETs, the scales of different RRMTs are often different, this WSM technique is not really suitable in MANETs.

4.4.3 The Weighted Product Model (WPM) Technique

The WPM [BRI22, MIL69] method is very similar to WSM. The main difference is that instead of using addition in the model, WPM uses multiplication. Each routing candidate is compared with the others by multiplying a number of ratios, one for each RRMT. Each ratio is raised to the power equivalent to the RRMTW of the corresponding RRMT. In
general, to compare two routing candidates, e.g. RC₁ and RC₂, the ratio between RC₁ and RC₂ has to be calculated:

\[
WPMR^A(\text{RC}_1/\text{RC}_2) = \prod_{j=1}^{n} \left(\frac{\text{RRMV}^j_1}{\text{RRMV}^j_2}\right)^{\text{RRMTW}^j_A}
\]

Equation 4.2

where \(WPMR^A(\text{RC}_1/\text{RC}_2)\) is a WPM ratio of RC₁ compare with RC₂ for application data type A. \(n\) is the number of RRMTs supported, \(\text{RRMV}^j_1\) is the metric value of RRMT\(_j\) for route RC₁, and \(\text{RRMTW}^j_A\) is the weight of RRMT\(_j\) for application data type A. Here, \(\text{RRMV}^j_1\) and/or \(\text{RRMV}^j_2\) must not be 0. Otherwise the relative value can become an infinity value (e.g. 0/0, or 0). In the case that \(\text{RRMV}^j_1\) is 0, a value of 1 will be used instead (i.e. use 1 to replace 0).

If \(WPMR^A(\text{RC}_1/\text{RC}_2)\) is greater than 1.0, it indicates that RC₁ is better or more desirable than RC₂, and vice versa. In the case that \(WPMR^A(\text{RC}_1/\text{RC}_2)\) is higher than 1.0, RC₂ will be dropped as it is believed to be worse than RC₁. Then RC₁ will be compared with the rest of the routing candidates. If the value of WPMR is equal to 1.0, then both routes are believed to be equally good. However, in this case, a routing candidate which is registered to the routing table first should be selected\(^1\). After comparing all the routing candidates against each other and dropping the worse ones, we will be able to select the best route.

Recall the Example Scenario 1, we make a routing decision using the WPM technique. The ratios between routing candidates are derived as follows:

\[
WPMR^A(\text{RC}_1/\text{RC}_2) = (25/10)^{0.20} \times (20/30)^{0.15} \times (15/20)^{0.40} \times (30/30)^{0.25}
\]

\[
= 1.007 > 1
\]

The value of \(WPMR^A(\text{RC}_1/\text{RC}_2)\) is higher than 1 which means RC₁ is better than RC₂. So RC₂ is dropped, then we compare RC₁ against RC₃, we get:

\[
WPMR^A(\text{RC}_1/\text{RC}_3) = 1.067 > 1
\]

\(^1\) This is because a RREP packet sent through a less delay route will arrive before a RREP packet sent through the route with a longer delay. After nodes receives a RREP packet, they will register a routing candidate (a list of intermediate nodes contained in the RREP) into a routing table.
$WPMR^A(RC_1/RC_3)$ is higher than 1.0. This shows that $RC_1$ is also better than $RC_3$. Therefore, the best route is $RC_1$, since it is superior to all the other routing candidates.

For a RRMT in a minimisation case, a lower RRMV may be considered as the better value than a higher RRMV (excluding a RRMT with a minus value). We use delay metric type as an example. A route with a lower delay is better than a route with a higher delay. The RRMTW value of this delay metric type will be multiplied by -1. Then the rest of the calculation can remain the same. This will allow us to use a RRMT such as delay metric type in our FRD framework.

Due to the use of relative values, WPM is more suitable to use in MANET than the WSM technique. This is because using a relative value allows us to use with multi-dimensional criteria. This means that even if different RRMTs have a different scale, this technique can still be applied to make a routing decision in MANETs.

4.4.4 The Analytic Hierarchy Process (AHP) Technique

The AHP [SAA80] technique was evolved from WSM. Instead of using an actual RRMV value to find a performance value and WSM, AHP uses a relative AHP value to find an AHP Score (AHPS). With the same RRMT, the relative $AHP_i^j$ value is calculated from the actual $RRMV_i^j$ of the $RC_i$ divided by the sum value of actual RRMV values of all routing candidates (from $RC_1$ to $RC_m$). That is,

$$AHP_i^j = \frac{RRMV_i^j}{\sum_{i=1}^{m}(RRMV_i^j)}$$  \hspace{1cm} \text{Equation 4.3}

A relative AHP value will then be summed up with other relative AHP values with different RRMTs of the same route. The total AHPS of $RC_i$ can be calculated by the following equation.

$$AHPS_i^A = \sum_{j=1}^{n}\left(\frac{RRMV_i^j}{\sum_{i=1}^{m}(RRMV_i^j)}\right) \times RRMTW^j_A$$  \hspace{1cm} \text{Equation 4.4}

Similar to the WSM technique, a routing candidate with the highest AHPS is considered as the best route. Here, we use the same example, i.e. Example Scenario 1, to explain how the AHP technique can be used to make a decision. The decision table is shown in Table 4.4.
Routing Candidates (RC_i) | RRMTs (RRMTW_A)
--- | --- | --- | --- | ---
RC_1 | 25/65 | 20/55 | 15/65 | 30/65
RC_2 | 10/65 | 30/55 | 20/65 | 30/65
RC_3 | 30/65 | 5/55 | 30/65 | 5/65

Table 4.4: A Decision Table of the Example Scenario 1 when using the AHP Technique

The AHP scores of the three routing candidates are:

\[
AHPS_1^A = \frac{25}{65} \times 0.20 + \frac{20}{55} \times 0.15 + \frac{15}{65} \times 0.40 + \frac{30}{65} \times 0.25 = 0.34
\]

\[
AHPS_2^A = \frac{10}{65} \times 0.20 + \frac{30}{55} \times 0.15 + \frac{20}{65} \times 0.40 + \frac{30}{65} \times 0.25 = 0.35
\]

\[
AHPS_3^A = \frac{30}{65} \times 0.20 + \frac{5}{55} \times 0.15 + \frac{30}{65} \times 0.40 + \frac{5}{65} \times 0.25 = 0.31
\]

Therefore, the best route for application data type A when using AHPS is route RC_2 because it has the highest AHPS of 0.35.

The sum of all relative AHP values for all the routing candidates in the same RRMT is equal to 1.0. For example, from Table 4.4, the total relative AHP values of RRMT_1 is 1 (i.e. \(AHPS_1^1 = \frac{25}{65} + \frac{10}{65} + \frac{30}{65}\)). This technique assigns a relative AHP value to each routing candidate depending on how high the RRMV value is compared to the RRMVs of other routing candidates. A route with a higher RRMV value will result in higher relative AHP value. Once a route has a higher relative AHP value, it may have a bigger chance to have the highest AHPS (AHPS is the sum of all relative AHP values). As a result, the route will have a higher chance to be selected as the best route.

### 4.4.5 The Revised Analytic Hierarchy Process (RAHP) Technique

As the name suggested, RAHP is a revised version of the original AHP model. It is proposed by Belton and Gear [BEL83]. They demonstrated that a ranking inconsistency can occur when AHP is used. According to the proposers, the inconsistency issue is caused by the fact that all relative AHP values for each RRMT are summed up to the value of 1.0. We use Example Scenario 2 (shown below) to illustrate this [BEL83].
Routing Candidates (RC<sub>i</sub>)  | RRMTs (RRMTW<sub>A</sub>)
--- | --- | --- | ---
RC<sub>1</sub> | 1/11 | 9/11 | 8/18
RC<sub>2</sub> | 9/11 | 1/11 | 9/18
RC<sub>3</sub> | 1/11 | 1/11 | 1/18

Table 4.5: A Decision Table of the Example Scenario 2 when using the AHP Technique

In this example, the AHPS of RC<sub>1</sub>, RC<sub>2</sub> and RC<sub>3</sub> are 0.45, 0.47 and 0.08, respectively. The RC<sub>2</sub> is the first rank. RC<sub>1</sub> is the second rank and the RC<sub>3</sub> is the third rank. So RC<sub>2</sub>, which is the best route among the three, will be chosen.

Assuming we add another routing candidate, RC<sub>4</sub>, to Example Scenario 2. Here, we call this new scenario as Example Scenario 3. The decision table for Example Scenario 3 has been revised as follows.

Routing Candidates (RC<sub>i</sub>)  | RRMTs (RRMTW<sub>A</sub>)
--- | --- | --- | ---
RC<sub>1</sub> | 1/20 | 9/12 | 8/27
RC<sub>2</sub> | 9/20 | 1/12 | 9/27
RC<sub>3</sub> | 1/20 | 1/12 | 1/27
RC<sub>4</sub> | 9/20 | 1/12 | 9/27

Table 4.6: A Decision Table of the Example Scenario 3 value when using the AHP Technique

Similar to the above, it can be verified that the AHPS of RC<sub>1</sub>, RC<sub>2</sub>, RC<sub>3</sub> and RC<sub>4</sub> are 0.37, 0.29, 0.06, 0.29, respectively. Then the ranking will be revised as follows: RC<sub>1</sub> is the first rank. RC<sub>2</sub> and RC<sub>4</sub> are second (as they both have the same AHPS), and RC<sub>3</sub> is the last rank. Belton and Gear [BEL83] claimed that this result is in logical contradiction with the previous result (RC<sub>2</sub> was the first and RC<sub>1</sub> was the second rank). As a result, they propose the revised version of AHP (RAHP).

In RAHP, instead of having a relative AHP value of all routing candidates sum up to one, the relative RAHP value is calculated from the RRMV value divided by the maximum RRMV value of all routing candidates in the same RRMT (i.e. max<sub>1<i<m</sub>(RRMV<sub>i,j</sub>)). The following equation finds the RAHP Score (RAHPS) of each routing candidate for application data type A,

\[
RAHPS^A_i = \frac{\sum_{j=1}^{n} \left( \frac{RRMV_{i,j}}{\max_{1<i<m}(RRMV_{i,j})} \right) \times RRMTW_j}{\sum_{j=1}^{n} RRMTW_j} \quad \text{Equation 4.5}
\]
where \( \max_{1<i<m} (RRMV_i^j) \) is the maximum RRMV of RRMT\(_j\) for all routing candidates (RC\(_1\), RC\(_2\), RC\(_3\), ..., RC\(_m\)). Now we use the Example Scenario 3 as an example to find the best route using the RAHP technique. The decision table is shown below.

<table>
<thead>
<tr>
<th>Routing Candidates (RC(_i))</th>
<th>RRMTs (RRMT(_W))</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( RRMTW^1_{A}(1/3) )</td>
</tr>
<tr>
<td>RC(_1)</td>
<td>1/9</td>
</tr>
<tr>
<td>RC(_2)</td>
<td>1 (9/9)</td>
</tr>
<tr>
<td>RC(_3)</td>
<td>1/9</td>
</tr>
<tr>
<td>RC(_4)</td>
<td>1 (9/9)</td>
</tr>
</tbody>
</table>

Table 4.7: A Revised Decision Table of the Example Scenario 3 when using the AHP Technique

The RAHPS of RC\(_1\), RC\(_2\), RC\(_3\) and RC\(_4\) are 2/3, 19/27, 1/9 and 19/27 respectively. It can be seen that four routing candidates are ranked as follows. RC\(_2\) and RC\(_4\) are the first rank. RC\(_1\) is the third rank, and RC\(_3\) is the forth rank. In this case, either RC\(_2\) or RC\(_4\) can be selected as the best route as they have the highest RAHPS.

For a minimisation RRMT (lower RRMs are better e.g. delay metric type), we can use the following equation to calculate the relative RAHP for this particular RRMT.

\[
\text{Relative RAHP} = \left( \frac{\min_{1<i<m} (RRMV_i^R)_{RRMT}}{RRMV_i^R_{RRMT}} \right) \times RRMTW_{RRMT}
\]

where \( \min_{1<i<m} (RRMV_i^R_{RRMT}) \) is the minimum value (the best value) of all routing candidates in terms of the RRMT. Then the RAHPS calculation technique will be the same as shown in Equation 4.4. With this modification, the RAHP technique will be able to accommodate the minimisation type of RRMT.

The RAHP technique was introduced to address the problem of result inconsistency in the original AHP technique. The results of the Example Scenario 2 and Example Scenario 3 show that RAHP can give a more consistent decision than the original AHP. Therefore, in the remaining part of this chapter we will only refer to RAHP (rather than AHP) in our discussion.
4.4.6 Further Discussions

We have seen several MCDM techniques that can be applied to routing decision making in MANETs. These are WSM, WPM, AHP and RAHP. One of these should be selected to use in the FRD framework. Table 4.8 shows the comparison of these four techniques.

<table>
<thead>
<tr>
<th></th>
<th>WSM</th>
<th>WPM</th>
<th>AHP</th>
<th>RAHP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using Relative Value</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Support Multi-dimensional Criteria</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Ranking Inconsistency Issue</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

*Table 4.8: Comparison Table between WSM, WPM, AHP and RAHP*

From the table above, we can see that WSM uses the actual RRMVs to calculate the performance value while other techniques use a relative value to do so. WSM requires all RRMTs to be in the same scale (i.e. single dimensional). Otherwise the performance value can be inaccurate when RRMTs are on different scale. As a result, the routing decision can also be inaccurate because the total score is inaccurate (as the total score is calculated from the performance values). For example, in a multi-dimensional case, the maximum values of RRMT$_1$ and RRMT$_2$ are 100 and 1000 respectively. If $RRMV^1_i$ and $RRMV^2_i$ have the same value of 100, the meaning of both values are totally different. For $RRMV^1_i$, the value of 100 means the perfect score (i.e. 100/100) while $RRMV^2_i$ means just 1/10 of the maximum score (i.e. 100/1000). If WSM is used in this situation, the performance value of both $RRMV^1_i$ and $RRMV^2_i$ are the same. Then a WSMS (Weighted Sum Model Score) will also be inaccurate. Thus, we cannot use the WSM techniques directly in our FRD framework design due to a variety of RRMT scales. To use the WSM technique, a normalisation mechanism is needed to convert RRMVs of multiple metric types into the same scale.

In contrary to WSM, WPM, AHP and RAHP use relative values. This means that, with these techniques, multiple RRMTs do not need to be at the same scale. Using the previous example, the maximum values of RRMT$_1$ and RRMT$_2$ are 100 and 1000 respectively. RC$_1$ has $RRMV^1_1$ of 30 and $RRMV^2_1$ of 700. RC$_2$ has $RRMV^1_2$ of 70 and $RRMV^2_2$ of 300. Comparing RC$_1$ with RC$_2$ using WPM, the relative value of the RRMT$_1$ and RRMT$_2$ are 30/70, and 700/300 respectively. Then these relative values will be used to find a final ratio/score. Therefore, multi-dimensional criteria are more suitable with these relative values than real performance values.
The FRD framework allows one of these MCDM to be plugged in to support multi-metric routing decision-making. In our discussions next, we will use WPM and RAHP. We do not select WSM and AHP because WSM requires a normalisation technique and AHP has the ranking inconsistency issue. The next section describes the FRD framework in details.

4.5 FRD Framework

The FRD framework is a framework designed to find the best route for an individual application. The FRD framework uses a cross layer approach which bridges the application layer and the network layer. It also makes use of a Multi-Criteria Decision Making (MCDM) technique to make a routing decision based on the application-layer requirements and routing information from the network layer.

4.5.1 FRD Overview

The FRD framework consists of 5 components: Application, RRMTs, RRMTWs, RRMVs and a MCDM technique.

![Figure 4.1: The components of the FRD framework](image)

To begin with, the first component, Application, is an application that is going to use the function provided by the FRD framework. One application may have different data types. Since, different application data types may have different QoS requirements, each different application data type can have their own choices, in terms of RRMTs, RRMTWs, and MCDM techniques, and use the chosen components to find the best route for themselves.
Relevant Routing Metric Types (RRMTs) are routing metric types chosen by an application. This choice is typically determined by the application requirements. For example, a source node may have two application data types running on the device, application data type A and application data type B. For application data type A, ETX and trust metric types are important, thus are chosen as the metric types to support its requirements. So RRMTs of application data type A are ETX and Trust Metric (TM) types. On the other hand, for application data type B, ETX and Remaining Battery Metric (RBM) types are important to support its requirements. So RRMTs of application data type B are ETX and RBM types. Here, we define the term of a RRMT domain as a union of all sets of RRMTs that a node (i.e. the source node) requires for all applications. In this case, the RRMT domain is a union of a set of RRMTs for application data type A and set of RRMTs for application data type B. The domain contains ETX, TM and RBM types.

Relevant Routing Metric Type Weights (RRMTWs) are weights assigned to each of the RRMTs associated to an application data type. They indicate how important each RRMT is. The higher the weight, the more important the RRMT is. For the same RRMT used by two different application data types, the assigned RRMTWs may be different.

Relevant Routing Metric Values (RRMVs) are routing metric values of each RRMT in the domain. These values are derived at the network layer. Each routing candidate has its own RRMV for each RRMT. Different routing candidates are likely to have different RRMVs.

Multi-Criteria Decision Making (MCDM) technique is the main component of the FRD framework. It takes RRMTWs and RRMVs to make a routing decision. An application may choose its preferred MCDM technique to use. For example, application data type A may select WPM while application data type B selects RAHP.

Figure 4.2 shows two application data types running on the device, application data type A and application data type B, along with their chosen metric types, metric type values and MCDM techniques. As mentioned earlier, each application can select its own MCDM technique to use.
4.5.2 Detailed FRD framework Description

A step-by-step procedure of how the FRD framework makes a routing decision is shown in Figure 4.3. There are 5 steps in total. They are: (1) selecting RRMTs and a MCDM technique, (2) assigning RRMTWs, (3) sharing RRMT domain, (4) sharing RRMVs and (5) making a routing decision.

Step 1. Selecting a set of Relevant Routing Metric Type (RRMT) and a preferred MCDM technique for each application

Step 2. Assigning Relevant Routing Metric Type Weight (RRMTW) for each RRMT in a set of RRMT from previous step

Step 3. Notifying the RRMT domain to its neighbours

Step 4. Acquiring RRMVs

Step 5. Making a Routing Decision

Figure 4.3: Procedures of the FRD framework
To map application-layer requirements onto network layer routing criteria, one needs to decide a RRMT domain for the application data types run at a node. This RRMT domain contains the set of RRMTs, which is selected based on the application data types’ QoS requirements. Using Figure 4.2 as an example, the node supports two application data types, i.e. A and B. Application data type A has two routing requirements: low loss rate and high reliability route. So the set of RRMTs chosen for application data type A are ETX metric type (for the high throughput requirement) and TM type (for the high reliable requirement). Both of these RRMTs will be used to make a routing decision for application data type A. On the other hand, application data type B may prefer a route with a high throughput and longer availability. So the set of RRMTs chosen for application data type B are the ETX metric type and RBM type.

![Figure 4.4: RRMT Domain of a Node Running Application Data Type A and Application Data Type B](image)

To summarise, the RRMT domain used by this node is shown in Figure 4.4. The RRMT domain contains three metric types, ETX, TM and RBM. The ETX and TM types are used by application data type A, and the ETX and RBM types are used by application data type B. The node will notify its neighbours of this RRMT domain at Step 3.

In addition to selecting RRMTs, the node (i.e. the user of the node) should also select a MCDM technique for each application. As mentioned above, the selected MCDM technique could be WSM, WPM, AHP, or RAHP; or it could be a future or emerging...
MCDM technique. The FRD framework is designed to be flexible in supporting the use of any metric types as well as any MCDM technique.

**Step 2: Assigning weights to all RRMTs of each RRMT set.**

Once a set of RRMTs and the MCDM technique are chosen for an application data type, the importance of each RRMT should be decided. This can be done by assigning a weight to each RRMT. The RRMT Weight (or RRMTW) should be a value between 0.0 and 1.0, and the weight sum of RRMTWs from each set of RRMTs must be equal to 1. A RRMT with a higher weight means it is more important or higher priority than a RRMT with a lower weight one.

As shown in Table 4.9, different applications may have different set of RRMTW values. In application data type A, the ETX metric type is considered more important than any other RRMTs. So the value of $RRMTW^A_{ETX}$ is set to the highest at 0.75. This value is higher than the weight of TM type ($RRMTW^A_{TM}$ at 0.25). The value of $RRMTW^A_{RBM}$ is set to 0 because the RBM type is not chosen for application data type A. On the other hand, for application data type B, the value of $RRMTW^B_{ETX}$ is 0.5. This value is different from the value of $RRMTW^A_{ETX}$ set for application data type A; the RRMTW values can be set differently depending on the requirements of different applications. Application data type B also has the value of $RRMTW^B_{RBM}$ of 0.5. For application data type B, the weight of both ETX and battery remaining metric types are equal (i.e. 0.5), this means that for this application data type two metric types are equally important.

<table>
<thead>
<tr>
<th>Applications</th>
<th>Relevant Routing Metric Type Weights (RRMTWs)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ETX Metric Type</td>
</tr>
<tr>
<td>Application A</td>
<td>0.75</td>
</tr>
<tr>
<td>Application B</td>
<td>0.50</td>
</tr>
</tbody>
</table>

*Table 4.9: Deciding a Weight for each RRMT of all RRMT sets*

At this step, RRMTWs are decided by developer. It can be done by using trial and error method to find the best suitable for their applications.

**Step 3: Notifying the RRMT domain to its neighbours**

At this step, the node notifies its neighbours of its RRMT domain. This is to let other nodes know that these RRMTs will be used to make a routing decision. Here, we use an
on-demand routing approach as an example for the FRD framework. The RRMT details of the RRMT domain will be carried in a RREQ routing packet in its extension field. The function of the RREQ packet in FRD is the same as RREQ packet in AODV protocol. It will be sent out when the node wants to start a communication, when a route is broken (received a RRER packet) and when an active route lifetime\(^1\) is timeout.

The shorter of the active route lifetime, the fresher and more accurate routing information that the node can obtain. However, the shorter active route lifetime means more overheads to flood a RREQ to the network. A longer active route lifetime is more suited to a network which has less mobility or a fix topology. This is because a network with a fixed topology may change network condition slower than a higher mobility network. The default value of the active route lifetime in a practical AODV protocol is 10 minutes [POO10].

Figure 4.5 shows an example of the RREQ packet with the RRMT domain included. The number of RRMTs carried in the packet depends on the size of the RRMT domain (i.e. \(n\)). The value contained in each RRMT field indicates the name of that RRMT. Suppose each RRMT field is 8-bits long, then each RRMT name should be assigned with a 8-bit value. For example, 1 (i.e. 0000 0001) = HOP_COUNT, 2=ETX, 3=TM, etc. This means all the names of the metric types should be coded into a 8-bit value. In addition, all the nodes in the network should know this convention.

Figure 4.5: Format of RREQ packets when using the FRD framework

Once a neighbouring node receives the RREQ packet, it checks whether it can support\(^2\) each of the RRMTs indicated in the RREQ packet. If this node supports the RRMTs, it will forward the RREQ packet to its downstream neighbouring node. Then the algorithm to acquiring the Relevant Routing Metric Values (RRMVs) of these RRMTs will start. For

---

1 Active Route Lifetime refers to a time period that an established route will last for.
2 To support a metric type (i.e. RRMT) means to have an algorithm to acquire the metric value of the RRMT installed in the system.
example, if the RRMT is an ETX metric type, the node will start broadcasting probes to its neighbouring nodes.

However, if the node does not support the RRMT(s), it will report back to the source node after it receives a RREP packet from the destination node. This process is to avoid having inaccurate routing information which caused by a missing routing metric value of some nodes. This is further explained in the next step.

*Step 4: Acquiring RRMVs*

When the destination node generates a RREP packet, the node will attach a pair of values (indicating RRMT and RRMV) for each RRMT it receives from the RREQ packet (as shown in Figure 4.6). For example, if a RRMT field indicates “HOP_COUNT” metric type, then the corresponding RRMV field will be an ETX value. If there are n RRMTs contained in RREQ, there will be n pairs of RRMTs and RRMVs contained in the RREP packet.

The generated RREP packet will be sent back to the upstream node. Once the upstream node receives the packet, it will update the RRMVs carried in the packet and send it to next upstream node. Using a hop count metric type as an example, the destination node generates a RREP packet with a RRMV of zero for the hop count metric type. Once the intermediate node next to the destination node (i.e. the destination node’s upstream node) receives the RREP packet, it updates the RRMV to 1 and sends the RREP packet further to its upstream node. This process will continue until the RREP packet reaches the source node.

![Figure 4.6: Format of RREP packets when using the FRD framework](image_url)

As mentioned in the previous step, if an intermediate node does not support a RRMT, it will override the corresponding RRMV field of the unsupported RRMT with a "NULL" value. However, it will still update the other RRMV fields of the supported RRMTs. After updating the RREP packet, the node will forward the RREP on. Once an intermediate node
receives a RREP packet and detects a "NULL" value in a RRMV field, it will leave this RRMV field with the "NULL" value.

When the source node detects a "NULL" RRMV value, i.e. an unsupported RRMT, it will stop acquiring the RRMV of that RRMT to save nodes resources. In the case of all of the RRMV fields are NULL, a default routing metric type of distance vector or a hop count will be used.

*Step 5: Making a routing decision*

After obtaining RRMTWs from the application layer and RRMVs from the network layer, this step will use the MCDM technique chosen for the application data type to make a routing decision. As mentioned earlier, here we only use the WPM and RAHP techniques to illustrate the routing decision making process, so the following description will be on how WPM and RAHP techniques make a routing decision.

### 4.5.3 FRD Discussions

The FRD framework is designed to better satisfy an application data type’s QoS requirements by choosing the most appropriate route to route the application data type. In addition to supporting different requirements defined at the application-level, the multiple routing metric type/algorithim selection capability offered by the FRD framework can also help to improve routing performance in a network. If only a single routing algorithm is supported, a route that has the best metric value (e.g. least hop count) will always be chosen. As a result, that route will be used more often. This can lead to one of two problems. One is the “bottleneck” problem, i.e. a certain part of the network will be busier than other parts; it may even be congested. The other is the network partition problem. If intermediate nodes are battery-powered, such as the case in mobile ad hoc networks, the nodes consistently serving a busy route may quickly run out of their battery power, in which case, some connections will be broken causing network partitions. In either case, the routing performance will be affected. However, by using our FDR framework, as a routing decision is made per application, traffic can be better distributed to take different routes. The chances for these problems to occur are reduced.

When comparing the FRD framework with other existing techniques, the process of selecting RRMTs is actually similar to how we assign a flow label into an IPv6 packet. In
IPv6 [RFC2460], there is an option to assign a route to a designated traffic flow. Each intermediate node is pre-specified a next hop node for each traffic flow with a different flow label. All packets with the same flow label will be routed through the same next hop node. This makes the flow label technique a fast and efficient way to forward a packet. However, this technique is designed for a network with a predictable topology (e.g. no movements of nodes or in an infrastructure network). It can be hard to implement in MANETs which have dynamic network topologies. It would be difficult to pre-define the next hop node when nodes in the network roam around, leaving, or joining in, the network dynamically.

The next section discusses how the simulation studies are carried out, and analyses the results from the study.

4.6 A Simulation Study

This section investigates the performance of the FRD framework and compares it against a number of traditional routing algorithms. These traditional routing algorithms are the Delay Aware Algorithm, Trust Aware Algorithm and Remaining Battery Aware Algorithm. For the FRD framework, two MCDM techniques, WPM and RAHP, are used. The investigation is carried out based on two scenarios, Scenario 4.1 and Scenario 4.2. Scenario 4.1 assumes that the conditions of all the intermediate nodes are perfect. Scenario 4.2 sets various conditions on the intermediate nodes. Both scenarios use the same network topology. The purpose of the first scenario is to investigate what the routing decision is like when the nodes are in a perfect condition, and the purpose of the second scenario is to investigate how the routing decision differs when the conditions of intermediate nodes change.

4.6.1 Investigation of Scenario 4.1

4.6.1.1 Scenario Description

This scenario simulates a biosensor ad hoc network. The network consists of a patient (source node or node 0), other mobile users (intermediate nodes or node 1-15) and an information centre (a destination node or node 16). The patient is equipped with a medical
sensor network on his body. A gateway connected to the sensor network periodically collects health and medical data from the sensors in the sensor network carried by the patient. The services provided to the patient range from regular monitoring of heartbeat rate to emergency calls (e.g. in case the heartbeats stop or are lower than a threshold). The data collected by the gateway is sent to an information centre using an ad hoc network.

There are three different types of data: (1) Delay sensitive Data Type (e.g. a life-threatening medical condition alert), (2) Reliability sensitive Data Type (e.g. sensitive and private data about the patient), and (3) General application Data Type (e.g. routine data monitored by the sensors).

(1) Delay sensitive data type: an example of this type of data can be a medical condition alert, to alert the information centre about a life-threatening condition of the patient. Other exemplar applications that generate similar data types include a disaster warning system or a real-time application such as video conference, voice over IP (VoIP) and online computer games. For a medical condition and disaster alert applications [PAZ08], data should be transferred to the information centre as soon as possible, and as reliably as possible. A route with the least delay is the most preferable one for such applications. Similarly, a real-time streaming application like a video conference or voice over IP also needs a route with a low delay.

For the delay sensitive data type, a routing metric type like a delay metric type can be really important. The delay metric type can help to identify which route can deliver data with the least delay. In addition, the delay sensitive data type also requires a good packet delivery ratio to avoid an interruption of the communication. So delay metric and trust metric types will be included in the RRMT set for this application data type. The data rate of this application is 100 packets/second for our simulation.

As mentioned in Section 4.5, to use the FRD framework, a weight should be assigned to each RRMT. Here, two RRMTs are used, the delay metric and trust metric types, and they are equally important to the delay sensitive data type, so the weights of these two RRMTs are 0.5 each \(RRMT_{DM}^{W_{DT}}\) is 0.5 and \(RRMT_{TM}^{W_{DT}}\) is 0.5, where DM stands for delay metric type, TM stands for trust metric type and
DDT stands for the delay sensitive data type). These values are set as an example to illustrate the working of our FRD framework. These parameter values are configurable depending on the requirements of the application or data type supported.

(2) Reliability sensitive data type: examples of applications that generate this data type can be sensitive and private data of patients, e.g. a patient’s name, contact details, or his medical history. This data type may not require a low delay route. Instead, it may require the most reliable route. A route with a higher packet delivery ratio can more reliable than a route with lower packet delivery ratio. So the trust metric is chosen as the most important routing metric type for this data type. In addition, the remaining battery metric type is also important to maintain the availability of the network as long as possible. So both trust metric and remaining battery metric types will be included into the RRMT set for the reliability sensitive data type. The data rate of this data type is set to 10 packets/second.

To assign RRMTWs for the reliability sensitive data type, the weight of the trust metric is higher than that of the remaining battery data type. This is because the main routing requirement of reliability sensitive data type is to search for the most reliable route. A route with a higher reliability should be given the highest priority. In this simulation, the weight of trust metric type (i.e. \(RRMTW_{RDT}^{TM}\), where RDT stands for reliability sensitive data type) is set to 0.75, and the weight of the remaining battery metric type (i.e. \(RRMTW_{RDT}^{RBM}\) where RBM stands for remaining battery metric type) is 0.25.

(3) General application data type: an example of this type of application data type could be the data from environmental monitoring. The monitored data should be sent to an information centre regularly and promptly. In such a case, the general application data type may consider all the RRMTs equally important. That means that the weights of the three routing metric types are equal, i.e. \(RRMTW_{GDT}^{DM}\), \(RRMTW_{GDT}^{TM}\) and \(RRMTW_{GDT}^{RBM}\) are 1/3 each (GDT stands for general application data type). The data rate of this application is set to 50 packets/second.
Table 4.10: RRMTW Table for Scenario 4.1

Table 4.10 summarises the RRMTW assignments for the application data types. If an application does not take the RRMT into consideration, the weight of the unconsidered RRMT will be 0 (so $RRMTW_{RBM}^{DDT}$ is 0 as shown in the table). The RRMT domain of this scenario contains delay metric, trust metric and remaining battery metric types.

The network topology of this scenario is shown in Figure 4.7. All the nodes start with the same condition, i.e. the trust metric value is 1 and the remaining battery metric values are 100%. There is no malicious node in the network. The only difference between these routes is the number of intermediate nodes en-route. Each route has a different number of intermediate nodes and the distance between nodes also varies. This will be further explained in the following paragraph.
As shown in Figure 4.7, the scenario consists of 17 nodes. The network space of the scenario is 200x200 meters\(^2\). The network topology is shown in the format of (x, y). The source node (node 0) is fixed at the position (0, 100) on the left-hand side of the network space. The destination node (node 16) is fixed at the position (200,100) on the right-hand side. The rest of the nodes are intermediate nodes. There are three routing candidates (i.e. available routes) between node 0 and node 16. These are Routing Candidate A (RC\(_A\)), Routing Candidate B (RC\(_B\)) and Routing Candidate (RC\(_C\)). RC\(_A\) is connected through wireless links with the maximum signalling data rate of 802.11g standard (at 54 Mbps). However, since the distances between are intermediate nodes are high, the actual signalling data rate will be lower than the maximum signalling data rate. This is because nodes will use a lower signalling data rate to increase a stability of a link. RC\(_A\) consists of nodes 0→4→7→10→13. RC\(_B\) has the lowest maximum data rate of 1 Mbps. RC\(_B\) consists of 0→1→3→6→9→11→13. RC\(_C\) is connected through a route, 0→2→5→8→12→13, with the maximum data rate of 11 Mbps. The intermediate nodes in the same route do not have a wireless link connect to any other nodes apart from the nodes in the same route as itself. For example, nodes in RC\(_A\) cannot connect to nodes in RC\(_B\) directly.

### 4.6.1.2 Route Selections

This section discusses how different routing algorithms make routing decisions. In our simulation, the FRD framework is investigated and compared with three traditional routing algorithms, delay, trust and remaining battery aware algorithms. The FRD framework employs two MCDM techniques, WPM and RAHP, respectively. Three application data types are used in the simulation. These are delay sensitive, reliability sensitive and general application data types. In this section, we shall investigate if these routing algorithms will select a different route for a different application data type (with different QoS requirements).

**Traditional Routing Algorithm**

Making a routing decision using a traditional algorithm is straight-forward. They do not consider the QoS requirements of application data types. This means that, for different application data types, these algorithms will select the same route (i.e. make the same routing decision) merely based on the network-layer routing metric values.
Making routing decision using the Delay Aware Algorithm

The delay aware algorithm selects a route that has the least delay. In this case, we use the delay metric type to measure the delay. The delay metric type is the elapsed time it takes for a routing packet to travel from the source node to the destination node and come back to the source node. In this case, we measured the delay metric value for each routing candidate and the results are shown in Table 4.11.

<table>
<thead>
<tr>
<th>Routing Candidate</th>
<th>Delay Metric Value (Seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RC_A</td>
<td>0.70</td>
</tr>
<tr>
<td>RC_B</td>
<td>3.06</td>
</tr>
<tr>
<td>RC_C</td>
<td>1.72</td>
</tr>
</tbody>
</table>

Table 4.11: The Delay Metric Values of all routing candidates

From the table, we can see that RC_A has the least delay. This result is as we expected as RC_A has the highest bandwidth and also the least hop count. So RC_A is selected by the delay aware algorithm.

Making routing decision using the Trust Aware Algorithm

The trust aware algorithm makes a routing decision based on the trust metric values of routing candidates. A routing candidate with the highest trust metric value will be selected by the trust aware algorithm. However, for this scenario, all the routing candidates have the same trust metric value (i.e. 1.0). According to the protocol specification in [XUE04], if there are two routes sharing the same highest trust metric value, a routing candidate with the least hop count will be selected (less wireless interference chance and less delay). That means the trust aware algorithm selects RC_A as its routing decision.

Making routing decision using the Remaining Battery Aware Algorithm

As all the routing candidates have the same remaining battery level (100%), the remaining battery aware algorithm will select a routing candidate with the least hop count. This is because the fewer hops, the less transmissions. Less transmissions mean less energy consumption. In this case, RC_A which has the least hop count is selected by the remaining battery aware algorithm.
**FRD Frameworks**

In contrast to the traditional routing algorithms mentioned above, the FRD framework takes into account application data types (their QoS requirements) when making a routing decision. Therefore, different application data types may be delivered along different routes. Since we consider 3 application data types, the FRD framework will make a routing decision three times, one for each application data type. Here, we first discuss how the FRD framework uses the WPM technique to make a routing decision.

**Making routing decision using the WPM Technique**

To make a routing decision using the WPM technique for the delay sensitive data type, we use the RRMTWs as shown in Table 4.10. The RRMTWs of delay metric, trust metric and remaining battery metric types are 0.5, 0.5 and 0 respectively. Using Equation 4.2, $RC_A$ is compared with $RC_B$ first. Here we get,

\[
WPMR^{DDT}(RC_A/RC_B) = (0.7/3.06)^{-0.5} \times (1/1)^{0.5} \times (100/100)^0
\]

\[
= 2.09
\]

The $WPMR^{DDT}(RC_A/RC_B)$ is higher than 1.0. This means that the $RC_A$ is better than $RC_B$. This result is obvious as the $RC_A$ has a better delay metric value than $RC_B$ when the trust and remaining battery metric values are all the same. Similarly, we also compare $RC_A$ with $RC_C$. The result is,

\[
WPMR^{DDT}(RC_A/RC_C) = (0.7/1.72)^{-0.5} \times (1/1)^{0.5} \times (100/100)^0
\]

\[
= 1.56
\]

From these results, it can be seen that $RC_A$ is also better than $RC_C$. Therefore, for the delay sensitive data type when using the WPM technique, $RC_A$ will be selected.

Since reliability sensitive data type has different RRMTW values (application layer requirements) from RRMTWs of the delay sensitive data type, we need to make a routing decision for reliability sensitive data type separately to the delay sensitive data type. The RRMTWs of delay, trust and remaining battery metric type of reliability sensitive data type are 0, 0.75 and 0.25 respectively. Here, $RC_A$ is compared with $RC_B$,
\[ WPMR^{RDT}(RC_A/RC_B) = (0.7/3.06)^0 \times (1/1)^{0.75} \times (100/100)^{0.25} \]

\[ = 1 \]

The \( WPMR^{RDT}(RC_A/RC_B) \) value is 1.0. This means \( RC_A \) and \( RC_B \) are both good for the reliability sensitive data type. However, as mentioned above, if the \( WPMR \) is equal to 1, a routing candidate which is registered into the routing table first will be considered as better. This is because it has a routing candidate derived when a RREP packet is received. A RREP packet which travels through a route with less delay is likely to arrive at the source node before a RREP packet which travels through a route with longer delay. Here, \( RC_A \) has less delay than \( RC_B \), so it arrives before \( RC_B \). As a result, we will keep \( RC_A \) and discard \( RC_B \). We then compare \( RC_A \) with \( RC_C \), the \( WPMR^{RDT}(RC_A/RC_C) \) is also equal to 1.0. Using the same rule, \( RC_A \) which is registered first is considered as a better routing candidate than \( RC_C \). As a result, \( RC_A \) will be selected for reliability sensitive data type.

For the general application data type, the RRMTW values of delay, trust and remaining battery metric types are 1/3, 1/3 and 1/3 respectively. To compare \( RC_A \) with \( RC_B \) and \( RC_C \), we have,

\[ WPMR^{GDT}(RC_A/RC_B) = (0.7/3.06)^{-\frac{1}{3}} \times (1/1)^{\frac{1}{3}} \times (100/100)^{\frac{1}{3}} \]

\[ = 1.63 \]

\[ WPMR^{GDT}(RC_A/RC_B) = (0.7/1.72)^{-\frac{1}{3}} \times (1/1)^{\frac{1}{3}} \times (100/100)^{\frac{1}{3}} \]

\[ = 1.34 \]

As \( RC_A \) is the best among the three routing candidates, it will be selected for reliability sensitive data type. Here, despite the differences of RRMTWs of all the application data types, \( RC_A \) is selected when using WPM technique.
Making routing decision using the RAHP Technique

Making a routing decision by using the RAHP technique requires the use of the best metric values for each RRMT. Here, the best value for delay metric type is 0.7 second (the lower is the better), for trust metric type it is 1 and for remaining battery metric type is 100. Table 4.12, 4.13 and 4.14 show the decision table of the RAHP technique for delay sensitive, reliability sensitive and general application data type, respectively.

<table>
<thead>
<tr>
<th>Routing Candidates</th>
<th>RRMTs (RRMTW_{DDT})</th>
<th>RAHPS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RRMTW^{DM}(0.5)</td>
<td></td>
</tr>
<tr>
<td>RC_{A}</td>
<td>(0.7/0.7) x 0.50</td>
<td>1.00</td>
</tr>
<tr>
<td>RC_{B}</td>
<td>(0.7/3.06) x 0.50</td>
<td>0.61</td>
</tr>
<tr>
<td>RC_{C}</td>
<td>(0.7/1.72) x 0.50</td>
<td>0.70</td>
</tr>
</tbody>
</table>

Table 4.12: A Decision Table of RAHP for Delay Sensitive Data Type in Scenario 4.1

<table>
<thead>
<tr>
<th>Routing Candidates</th>
<th>RRMTs (RRMTW_{RDT})</th>
<th>RAHPS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RRMTW^{DM}(0)</td>
<td></td>
</tr>
<tr>
<td>RC_{A}</td>
<td>(0.7/0.7) / 3</td>
<td>1.00</td>
</tr>
<tr>
<td>RC_{B}</td>
<td>(0.7/3.06) / 3</td>
<td>0.74</td>
</tr>
<tr>
<td>RC_{C}</td>
<td>(0.7/1.72) / 3</td>
<td>0.80</td>
</tr>
</tbody>
</table>

Table 4.13: A Decision Table of RAHP for Reliability Sensitive Data Type in Scenario 4.1

<table>
<thead>
<tr>
<th>Routing Candidates</th>
<th>RRMTs (RRMTW_{GDT})</th>
<th>RAHPS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RRMTW^{DM}(1/3)</td>
<td></td>
</tr>
<tr>
<td>RC_{A}</td>
<td>(0.7/0.7) / 3</td>
<td>1.00</td>
</tr>
<tr>
<td>RC_{B}</td>
<td>(0.7/3.06) / 3</td>
<td>0.74</td>
</tr>
<tr>
<td>RC_{C}</td>
<td>(0.7/1.72) / 3</td>
<td>0.80</td>
</tr>
</tbody>
</table>

Table 4.14: A Decision Table of RAHP for General Application Data Type in Scenario 4.1

The results show that when using the RAHP technique, RC_{A} has the highest RAHPS for delay sensitive and general application data types. Thus, the routing decisions for delay sensitive and general application data types are RC_{A}. For the reliability sensitive data type, all the routing candidates have the same RAHPS. If we use the same rule (i.e. a routing candidate which is registered into the routing table first is the best) to make a decision in this case, RC_{A} is also the best decision for the reliability sensitive data type.

To summarise, in this scenario, all the routing algorithms have selected RC_{A}. This is because RC_{A} has the least delay and the least hop count while the other metric value (i.e. trust and remaining metric types) are the same as other routes.
The next section discusses how the simulation is setup. It explains how the parameter values are selected and how the existing protocol is modified to use in the simulation.

### 4.6.1.3 Simulation Setup

The simulation is done using NS2 [NS2] (the same simulation package as for Chapter 4). The main routing protocol used in this simulation study is the AODV-UU protocol [AODVUU]. As the study of the FRD framework is carried out with the use of three RRMTs, delay, trust and remaining battery metric types, this routing protocol is modified to consider these metric values during a routing discovery process.

For delay metric type, it selects a route that has the least round trip delay. The round trip delay is the difference between the time when RREQ is transmitted and when RREP is received by the source node. When this routing metric is used, a routing candidate which has the least round trip delay will be selected. These RREQ and RREP packets are implemented in the AODV-UU routing protocol by default.

For the trust metric type, a direct trust model is used. With this model, an intermediate node, after forwarding a packet to its neighbour, will ‘listen’ if the neighbour has actually forwarded the packet on. If it has not heard the packet forwarding after a timeout, it assumes that there is a packet dropping attack. This ‘listening’ task can be done by using the watchdog scheme [MAR00]. Based upon the monitored result, the node will derive a trust value for the neighbouring node. This value is carried in the RREP packet which is sent to the source node. The source node will use all the trust values returned from all the intermediate nodes along a route to calculate an average trust value for the route. The route with the highest average trust value will be selected during the routing decision making process (provided that the trust metric type is used).

The remaining battery metric type is used to find a route with the highest remaining battery level. When the energy model is enabled, the remaining battery information can be exchanged between nodes. The protocol is modified such that intermediate nodes append their respective remaining battery levels in the RREP packet. If an intermediate node has a remaining battery level higher than the one carried in the RREP packet, it will simply forward the packet on. Otherwise, it will replace the value of remaining battery level carried in the packet with its own value. When the source node receives the RREP packet,
the source node will receive the lowest remaining battery level of all intermediate nodes en-route. Based on the remaining battery level carried in the RREP packet, the source node can choose a route that has the highest remaining battery power. If two routes have the same remaining battery level then the route with the lower hop count will be selected, as a route with a lower hop count should consume less energy in forwarding the traffic.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>NS2 Version</td>
<td>2.26</td>
</tr>
<tr>
<td>Routing Protocol</td>
<td>AODVUU</td>
</tr>
<tr>
<td>Node Movement Speeds</td>
<td>0 meter/second</td>
</tr>
<tr>
<td>WLAN Standard</td>
<td>802.11g</td>
</tr>
<tr>
<td>Link Bandwidth</td>
<td>1, 36 and 54 Mbps</td>
</tr>
<tr>
<td>Nominal Radio Range</td>
<td>50 meters</td>
</tr>
<tr>
<td>Network Domains</td>
<td>200 x 200 meters</td>
</tr>
<tr>
<td>Signal Strength Reception</td>
<td>7.69113e-08</td>
</tr>
<tr>
<td>Carrier Sensing Threshold</td>
<td>7.69113e-08</td>
</tr>
<tr>
<td>Data Rate (packets/second)</td>
<td>10, 50 and 100</td>
</tr>
<tr>
<td>Buffer Queue Length</td>
<td>1,000 packets</td>
</tr>
<tr>
<td>Traffic Type</td>
<td>UDP</td>
</tr>
<tr>
<td>Data Packet Type</td>
<td>CBR</td>
</tr>
<tr>
<td>Application Data Payload Size</td>
<td>800, 1000, 1200 and 1400 bytes/packet</td>
</tr>
<tr>
<td>Number of Mobile Nodes</td>
<td>17</td>
</tr>
<tr>
<td>Initial Energy</td>
<td>100 Joules</td>
</tr>
<tr>
<td>Energy Consumption in Transmitting Packets</td>
<td>1400 mW</td>
</tr>
<tr>
<td>Energy Consumption in Receiving Packets</td>
<td>1000 mW</td>
</tr>
<tr>
<td>Energy Consumption in IDLE State</td>
<td>830 mW</td>
</tr>
<tr>
<td>Energy Consumption in Sleep State</td>
<td>130 mW</td>
</tr>
</tbody>
</table>

Table 4.15: The parameters used in the simulation
The parameters used in this simulation study are summarised in Table 4.15. In detail, the dimension of space in the network (e.g. network area) is set as 200x200m². The traffic type of this scenario is Constant Bit Rate (CBR). The traffic is generated at the source node and transmitted to the destination node. The data rate used is dependent on the application data type, i.e. 100 packets/second for the delay sensitive data type, 10 packets/second for the reliability sensitive data type and 50 packets/second for the general application data type. The source and destination nodes are fixed during a simulation run. However, if the current route is broken (due to intermediate nodes dying), a new route will be re-established. This means a new route discovery process will be initiated. The sizes of the data packets are set to 800, 1,000, 1,200 and 1,400 bytes, respectively. The other parameter values remain the same as those used in the simulation study presented in the previous chapter, which have been summarised in Table 4.15.

4.6.1.4 Simulation Results

In the investigation, three performance metrics are used. These are Average Throughput (AT), Packet Delivery Ratio (PDR), and Route Breaking Time (RBT).

Route Breaking Time (or known as path duration [SAD03]) is used as a performance metric in Simulation 4.1. It measures the time elapsed from when a node starts communication to when the route is broken. The route breaking time shows how well the routing algorithm select the first route in terms of the longest availability without fail. There can be several reasons for a route to break, e.g. when an intermediate node runs out of battery, leaves the network, or switches off. Since the link is broken, a new route is needed to continue with the communication. If new routes need to be discovered very often, there will be lots of broadcasting traffic generated in the network and more delays will be experienced when discovering a route.

As the routing decisions made by all routing algorithms for this scenario are the same (i.e. \(RC_A\)), we will show the performance of \(RC_A\) and compare it with the performances of \(RC_B\) and \(RC_C\). In this way, we can see the differences in the performances of the three routing candidates. These results will also be used when examining the simulation results from Scenario 4.2 later on. The results are shown below.
Figures 4.8-4.10 show the average throughputs of the three application data types. They show that when the data rate is low (e.g. 10 packets/second as generated by the reliability sensitive application), the average throughput of the three routing candidates are not much different even though the bandwidths of the three routing candidates are different. This is because there is ample bandwidth to forward the traffic, and the network is not so busy to forward data packets along the route. However, when the data rate goes higher, routing candidates with a higher bandwidth provide higher average throughputs. RC_A and RC_C have higher average throughputs than RC_B for delay sensitive and general application data types as shown in Figures 4.9 and 4.10.

![Average throughputs of Scenario 4.1 for reliability sensitive data type (10 packets/second)](image)

Figure 4.8: Average Throughputs for Reliability Sensitive Data Type in Scenario 4.1
Average throughputs of Scenario 4.1 for general application data type (50 packets/second)

![Bar chart showing average throughputs for general application data type in Scenario 4.1. The chart compares Route A, Route B, and Route C for packet sizes ranging from 800 to 160K bytes.]

Figure 4.9: Average Throughputs for General Application Data Type in Scenario 4.1

Average throughputs of Scenario 4.1 for delay sensitive data type (100 packets/second)

![Bar chart showing average throughputs for delay sensitive data type in Scenario 4.1. The chart compares Route A, Route B, and Route C for packet sizes ranging from 800 to 160K bytes.]

Figure 4.10: Average Throughputs for Delay Sensitive Data Type in Scenario 4.1
Figure 4.11: Route Breaking Times for General Application Data Type in Scenario 4.1

Figure 4.11 shows that RC_A has the longest route breaking time. To investigate the reasons behind this observation, we conducted further simulations (Scenario 4.1.1) to see if different signalling rates would affect the length of the route breaking time. In Scenario 4.1.1, we used the same network topology as this scenario. We only investigated RC_A. The data packet size is set to 800 bytes, the data rate to 10 packets/second. Different signalling rates were investigated; they were 1, 11, 24, 36 and 54Mbps. The route breaking time results are given in Figure 4.12.
Figure 4.12 shows that a higher signalling data rate generally leads to a longer route breaking time. So we have also checked with the remaining battery levels when having different signalling data rates. The results are shown in Figure 4.13. It shows that in the
same condition, a slower signalling data rate consumes energy faster than a faster signalling data rate. The study in [WEN10] also shows similar results on the relationship between signalling data rate and power consumption.

![Packet Delivery Ratios of Scenario 4.1 for Reliability Sensitive Data Type (10 packets/second)]

Figure 4.14: Packet Delivery Ratios for Reliability Sensitive Data Type in Scenario 4.1

The packet delivery ratios of all the routing candidates are shown in Figures 4.14-4.16. The results show that no data or very little data is lost during their transmissions via routes RC_A and RC_C. However, for the route RC_B, more data are lost when the data rate is higher and when the packet size is bigger. For the reliability sensitive data type, the average packet delivery ratio of RC_B is around 97% (shown in Figure 4.14). The average packet delivery ratio of RC_B reduces to 23% when the data rate increases to 50 packets/second (shown in Figure 4.15 for the general application data type), and then it further reduces to 12% when the data rate increases to 100 packets/second (shown in Figure 4.16).

To investigate the data loss rate of route RC_B, we looked for how long packets are sent through each route. It turns out that route RC_B has the longest time to transmit each packet. For the same size of data packet (1,400 bytes), an average transmission delay of a link of RC_B (with the maximum data rate of 1Mbps at the optimal range) is around 0.013 second per packet. This is very slow when compared with the delay of links of route RC_A and RC_C, which are 0.001 and 0.002 seconds respectively. This means nodes on route RC_B
take much longer time to transmit a packet than the other routes. When the data rate is higher, the wireless interference problem is occurred more and more at RC_B.

Figure 4.15: Packet Delivery Ratios for General Application Data Type in Scenario 4.1

Figure 4.16: Packet Delivery Ratios for Delay Sensitive Data Type in Scenario 4.1

In this scenario, all routing algorithms make a good routing decision for all the application data types. This is because it is obvious that RC_A is the best route among the three candidates. However, when the condition of the network is different (e.g. with some malicious behaviours), different routing algorithms may select different routes. In the next
section, we investigate a scenario with a network condition that is very different from the one in Scenario 4.1.

4.6.2 Investigation of Scenario 4.2

4.6.2.1 Scenario Description

Scenario 4.2 has the same network topology as Scenario 4.1. It also has the same application data types: delay sensitive, reliability sensitive and general application data type. The main difference between this scenario and the previous one is that the conditions of intermediate nodes are different. The details of this scenario are shown as follows.

Here, node 3 is assumed to be malicious. It performs packet dropping attacks. It randomly drops data packets at a ratio of 1 in every 10 packets. Prior to joining the network, node 2 has a past experience with node 3. In order to see the impact of this malicious node clearly and simple, the trust value of this route is set to 0.1 before the route selection process begins.

In addition, node 14 is assumed to have been operating in other networks before joining in this network, so the remaining battery level of this node is lower than other nodes. Other nodes’ battery levels are all set to 100%, whereas node 14’s battery level is 95%. Therefore, this node may not last as long as the other nodes. Another setting for this node

![Figure 4.17: The topology of the network under study, Scenario 4.2](image-url)
is that it has a not-so-good track record in forwarding packets. In other words, the trust value of this node is rated at 0.95. This trust value may be slightly lower than the trust values of other nodes, but it is enough for this node to be excluded when making a routing decision by using the trust aware algorithm.

To summarise, in this scenario, $RC_A$ has the highest bandwidth (as in the previous scenario). However, this route contains a malicious node, so it has the lowest trust value. $RC_B$ has the lowest bandwidth. The hop count of $RC_B$ is also higher than other routes. This means that the chance of having wireless interference during transmission along this route is also higher than the other routes. However, $RC_B$ has the highest trust metric value. $RC_C$ is the route with performance metric values ranking between the other two routes. It has a medium level of bandwidth and a medium hop count (6 hops). The trust value of this route is not the highest level but it is very close to the highest one.

4.6.2.2 Route Selections

*Traditional algorithms*

The delay aware algorithm attempts to select a route with the least delay. As $RC_A$ has the least delay (as shown in Table 4.11), delay aware algorithm simply selects $RC_A$. However, once $RC_A$ is broken, another route should be selected. Here, we expect that the condition of the other routing candidates (i.e. $RC_B$ and $RC_C$) should not be too much different from the beginning of the simulation. The delay metric values of both routing candidates should be the same. The trust metric values should also be the same as they have not been selected to transmit a data packet. The question is would remaining battery metric value be the same? Since $RC_B$ and $RC_C$ are not used in forwarding a packet, the battery draining rate should be very similar. Therefore, the metric values we measured at the beginning of the simulation would still be valid. This means that the second preferable route will be selected once the first selected route is broken. In the case of the delay aware algorithm, $RC_C$, which has the second least delay, will be selected. Here, the routing decision made by the delay aware algorithm is denoted as $RC_A\rightarrow C$ as it has selected $RC_A$ first and then $RC_C$ after $RC_A$ has broken.

The trust aware algorithm attempts to select a route with the highest trust metric value. It selects $RC_B$ as the first choice, as $RC_B$ has the highest trust metric value. If $RC_B$ is broken
(e.g. an intermediate node dies), it will select $RC_C$ as $RC_C$ has a higher trust metric value than $RC_A$. Therefore, the routing decision made by the trust aware algorithm will be $RC_B \rightarrow C$.

The remaining battery aware algorithm attempts to select a route with the highest remaining battery level. $RC_C$ contains node 14 that has a lower remaining battery level (i.e. 95%), so $RC_C$ becomes less preferable for the remaining battery aware algorithm. $RC_A$ and $RC_B$ have the same battery level (i.e. 100%). Since $RC_A$ has a lower hop count, $RC_A$ will be selected first. If $RC_A$ is broken, $RC_B$ would be selected next as it has a higher remaining battery level than $RC_C$. The routing decision made by the trust aware algorithm will be $RC_A \rightarrow B$.

**FRD Framework using the WPM Technique**

For the delay sensitive data type, RRMTWs of delay, trust and remaining battery metric types are 0.5, 0.5 and 0, respectively (as shown in Table 4.10). We compare the three routing candidates, then we have,

$$WPMR^{DDT}(RC_A/RC_B) = 0.66$$

$$WPMR^{DDT}(RC_B/RC_C) = 0.77$$

The above results show that $RC_C$ is better than $RC_B$ and also $RC_B$ is better than $RC_A$, so $RC_C$ is the first choice and then $RC_B$ is the second choice for the delay sensitive data type. The routing decision for the delay sensitive data type is $RC_C \rightarrow B$.

For the reliability sensitive data type, RRMTWs of delay, trust and remaining battery metric types are 0, 0.75 and 0.25, respectively (shown in Table 4.10). The results are shown as follow.

$$WPMR^{RDT}(RC_A/RC_B) = 0.18$$

$$WPMR^{RDT}(RC_B/RC_C) = 1.05$$

As we can see, the $WPMR^{RDT}(RC_A/RC_B)$ value is 0.18. This means $RC_A$ is much worse than $RC_B$ for the reliability sensitive data type. This is obvious as the trust metric value of $RC_A$ is much lower than that of $RC_B$ (0.1 versus 1.0). In addition, the
$WPMR^{RDT}(RC_B/RC_C)$ value is higher than 1. It means $RC_B$ is a better route than $RC_C$. If we look at the RRMV’s of both routes, we can see that $RC_B$’s trust metric and remaining battery values are all better than $RC_C$’s. So $RC_B$ would be the first choice for the reliability sensitive data type.

If $RC_B$ is broken, another route should be sought. So we should find out which one of $RC_A$ and $RC_C$ is better. We compare $RC_A$ and $RC_C$.

$$WPMR^{RDT}(RC_A/RC_C) = 0.19$$

The result above shows that $RC_C$ is better than $RC_A$. So $RC_C$ will be selected as the second route after $RC_B$ is broken for the reliability sensitive data type. The full routing decision for the reliability sensitive data type is therefore $RC_B\rightarrow C$.

The RRMTW values of the general application data type are $1/3$ for all the RRMTs. We first compare $RC_A$ with $RC_B$ and then with $RC_C$,

$$WPMR^{GDT}(RC_A/RC_B) = 0.76$$
$$WPMR^{GDT}(RC_B/RC_C) = 0.85$$

From these calculations, it can be seen that $RC_B$ is better than $RC_A$, and $RC_C$ is also better than $RC_B$. So $RC_C$ is the first choice and $RC_B$ is the second choice for the general application data type. The routing decision for the general application data type is therefore $RC_C\rightarrow B$. That is, if $RC_C$ is broken, $RC_B$ will be selected.

FRD Framework using the RAHP technique

The first thing to do when using the RAHP technique is to find the best value for each RRMT. Here, the best delay metric value is 0.7 second, the best trust metric value is 1 and the best remaining battery metric value is 100. Similar to the previous scenario, the routing decisions are made after we derive a decision table for each data type. Tables 4.16, 4.17 and 4.18 are the decision tables used by the RAHP technique for delay sensitive, reliability sensitive and general application data type, respectively.
The decision table for the delay sensitive data type shows that the RAHPS of RC_C is the highest value, and that of RC_B is the second highest. So, a routing decision made by RAHP for the delay sensitive data type is RC_C→B.

For the reliability sensitive data type, it places the most focus on the trust value and some on remaining battery metric types. RC_B is obviously the best route as it has the highest RRMVs on both trust and remaining battery metric types. So RAHPS of RC_B is the highest, and RC_C has the second highest RAHPS. As a result, RC_B→C is the routing decision for the reliability sensitive data type.

When making a routing decision for the general application data type, all the RRMTs will be considered. Each RRMT are weighted equally (i.e. 1/3). From Table 4.18, we can see that the RAHPS of these three routes are quite close. This is because each of the routes has its own strengths and weaknesses. RC_A has the best delay metric value (i.e. at 0.7 second), but very weak at trust metric value (i.e. 0.1). RC_B has the best trust metric value (i.e. 1) and remaining battery metric value (i.e. 100) but weakest at delay metric value (i.e. 3.06 second). Since RC_C has good average values of all the RRMVs, the RAHPS of RC_C is the highest for the general application data type. For the second route choice, RC_B is selected.

---

<table>
<thead>
<tr>
<th>Routing Candidates</th>
<th>RRMTs ($RRMT_{DRT}$)</th>
<th>RAHPS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$RRMTW_{DM}(0.5)$</td>
<td></td>
</tr>
<tr>
<td>RC_A</td>
<td>(0.7/0.7) x 0.5</td>
<td>0.55</td>
</tr>
<tr>
<td>RC_B</td>
<td>(0.7/3.06) x 0.5</td>
<td>0.61</td>
</tr>
<tr>
<td>RC_C</td>
<td>(0.7/1.72) x 0.5</td>
<td>0.67</td>
</tr>
</tbody>
</table>

Table 4.16: A Decision Table of RAHP for Delay Sensitive Data Type in Scenario 4.2

<table>
<thead>
<tr>
<th>Routing Candidates</th>
<th>RRMTs ($RRMT_{RDT}$)</th>
<th>RAHPS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$RRMTW_{DM}(0)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$RRMTW_{TM}(0.75)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$RRMTW_{RBM}(0.25)$</td>
<td></td>
</tr>
<tr>
<td>RC_A</td>
<td>(0.7/0.7) x 0</td>
<td>0.33</td>
</tr>
<tr>
<td>RC_B</td>
<td>(0.7/3.06) x 0</td>
<td>1.00</td>
</tr>
<tr>
<td>RC_C</td>
<td>(0.7/1.72) x 0</td>
<td>0.95</td>
</tr>
</tbody>
</table>

Table 4.17: A Decision Table of RAHP for Reliability Sensitive Data Type in Scenario 4.2

<table>
<thead>
<tr>
<th>Routing Candidates</th>
<th>RRMTs ($RRMT_{GDR}$)</th>
<th>RAHPS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$RRMTW_{DM}(1/3)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$RRMTW_{TM}(1/3)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$RRMTW_{RBM}(1/3)$</td>
<td></td>
</tr>
<tr>
<td>RC_A</td>
<td>(0.7/0.7) / 3</td>
<td>0.70</td>
</tr>
<tr>
<td>RC_B</td>
<td>(0.7/3.06) / 3</td>
<td>0.74</td>
</tr>
<tr>
<td>RC_C</td>
<td>(0.7/1.72) / 3</td>
<td>0.77</td>
</tr>
</tbody>
</table>

Table 4.18: A Decision Table of RAHP for General Application Data Type in Scenario 4.2
as it has the second highest RAHPS. As a result, the routing decision for the general application data type is RC_{C→B}.

Table 4.19 shows a summary of the routing decisions made by different routing algorithms. Delay aware algorithm selects RC_{A→C} for all the application data types. Trust aware algorithm selects RC_{B→C} for all the application data types. Remaining battery aware algorithm selects RC_{A→B} for all the application data types. However, with the use of the FRD framework, different decisions are made for different application data types. For delay sensitive data type and general application data type, the FRD framework selects RC_{C→B}. For the reliability sensitive data type, the FRD framework selects RC_{B→C}.

<table>
<thead>
<tr>
<th>Application Data Type</th>
<th>Tradition Routing Algorithms</th>
<th>FRD Framework</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Delay Aware</td>
<td>Trust Aware</td>
</tr>
<tr>
<td>Delay sensitive Data Type (DDT)</td>
<td>RC_{A→C}</td>
<td>RC_{B→C}</td>
</tr>
<tr>
<td>Reliability sensitive Data Type (RDT)</td>
<td>RC_{A→C}</td>
<td>RC_{B→C}</td>
</tr>
<tr>
<td>General Data Type (GDT)</td>
<td>RC_{A→C}</td>
<td>RC_{B→C}</td>
</tr>
</tbody>
</table>

Table 4.19: Summary of Routing Decisions

4.6.2.3 Simulation Results

The legends, DAA, TAA and RBAA, indicate that routing decisions are made based upon the routing algorithms: Delay Aware Algorithm, Trust Aware Algorithm and Remaining Battery Aware Algorithm, respectively. Since in this scenario the routing decisions made by WPM and RAHP are the same, the legend of FRD means the routing decisions made using a MCDM technique (WPM or RAHP). The routing decision of the FRD framework for DDT, RDT and GDT are denoted as FRD/DDT, FRD/RDT and FRD/GDT, respectively.

This section shows and discusses the results collected from several simulation settings reflecting different application data types. The simulation results of the delay sensitive data type will be discussed first, then the results of the reliability sensitive data type, and lastly the results of the general application data type.
Delay Sensitive Data Type

Figure 4.18 and 4.19 show the average throughputs and packet delivery ratios collected from the simulation settings of the delay sensitive data type. Using the FRD framework (WPM and RAHP) to select routes (i.e. RC_{C\rightarrow B}) has actually led to a better level of average throughput and packet delivery ratio than using delay, trust or remaining battery aware algorithms alone. Making routing decisions using the delay aware algorithm (RC_{A\rightarrow C}) or the remaining battery aware algorithm (RC_{A\rightarrow B}) generates very low levels of average throughputs and packet delivery ratios. In both of these cases, RC_{A} was first selected. Since RC_{A} includes a malicious node en-route, most of the traffic transmitted through this malicious node was dropped (due to the black hole attack). So the packet delivery ratio of this route is significantly reduced. A lower packet delivery ratio will also lead to a lower average throughput. As a result, average throughputs and packet delivery ratios when using the delay aware algorithm and the remaining battery aware algorithm to make a routing decision are lower than the values when the FRD framework is used in all the cases investigated.

![Average Throughputs for Delay Sensitive Data Type in Scenario 4.2 (100 packets/second)](image)

**Figure 4.18:** Average Throughputs for Delay Sensitive Data Type in Scenario 4.2

178
A question arises here: why the packet delivery ratios of $RC_{A\rightarrow B}$ and $RC_{A\rightarrow C}$ are better than 10% when the malicious node en-route could only forward about 10% of the data packets. This is because when $RC_A$ is broken, $RC_B$ or $RC_C$ will be selected. As we know that $RC_B$ is not affected by any malicious node and $RC_C$ could deliver 95% of the data packets, the average packet delivery ratios of $RC_A$ plus $RC_B$ or $RC_A$ plus $RC_C$ will be higher than the packet delivery ratios of the $RC_A$ alone.

When using trust aware algorithm, the route selected ($RC_{B\rightarrow C}$) does not perform well in terms of average throughput and packet delivery ratio either. Although the trust metric value of the route selected by trust aware algorithm is the highest, the bandwidth capacity of the first route selected by trust aware algorithm ($RC_B$) was too low. It cannot support the data rate required by the delay sensitive data type well. A lot of data packets were dropped. As a result, average throughputs and packet delivery ratios are very low.

![Figure 4.19: Packet Delivery Ratios for Delay Sensitive Data Type in Scenario 4.2](image)

By comparing the results presented in Figures 4.18 and 4.19, it is clear that the FRD framework can make the best decision for the delay sensitive data type. The delay sensitive data type’s QoS requirements the delay sensitive data type are high throughput and high packet delivery ratio. The route selected by the FRD framework ($RC_{C\rightarrow B}$) can deliver the best of both average throughput and packet delivery ratio results.
Reliability Sensitive Data Type

Unlike the packet delivery ratios for the delay sensitive data type (Figure 4.19), Figure 4.20 shows that, for the reliability sensitive data type, the packet delivery ratios of the trust aware algorithm (RC_{B→C}) and the FRD framework (RC_{C→B}) are very close. This is because the data rate of the reliability sensitive data type is much lower than the delay sensitive data type (10 packets/second against 100 packets/second). With a low data rate, the bandwidth capacity of the route first selected by trust aware algorithm (RC_{B}) has not been fully used. As a result, both trust aware algorithm and FRD framework have made a good decision for the reliability sensitive data type in terms of packet delivery ratio.

The reliability sensitive data type also prefers a route with a good remaining battery level. In this case, the delay and the remaining battery aware algorithms select RC_{A→C} and RC_{A→B}, respectively. RC_{A} may have the highest bandwidth and the lowest hop count. A route with a lower hop count would use less energy to transmit a packet than a route with a higher hop count. Also we have learnt from Figure 4.12 that a route with a higher level of signalling data rate usually lasts longer than a route with a lower level of bandwidth. Therefore, RC_{A→B} and RC_{A→C} have the longest simulation duration as we have expected. This means that the decision made by using the delay and the remaining battery aware

---

**Figure 4.20: Packet Delivery Ratios for Reliability Sensitive Data Type in Scenario 4.2**

The reliability sensitive data type also prefers a route with a good remaining battery level. In this case, the delay and the remaining battery aware algorithms select RC_{A→C} and RC_{A→B}, respectively. RC_{A} may have the highest bandwidth and the lowest hop count. A route with a lower hop count would use less energy to transmit a packet than a route with a higher hop count. Also we have learnt from Figure 4.12 that a route with a higher level of signalling data rate usually lasts longer than a route with a lower level of bandwidth. Therefore, RC_{A→B} and RC_{A→C} have the longest simulation duration as we have expected. This means that the decision made by using the delay and the remaining battery aware
algorithms is the best in terms of simulation duration results. On the other hand, the trust aware algorithm has the lowest simulation duration. This result is the worst for the reliability sensitive data type in terms of the longest availability requirement. For the route selected by the FRD framework (RC_{C\rightarrow B}), the simulation durations of this route may not last as long as the routes selected by the delay aware algorithm (RC_{A\rightarrow C}) and the remaining battery aware algorithm (RC_{A\rightarrow B}), but it is much longer than the route selected by the trust aware algorithm (RC_{B\rightarrow C}).

![Simulation Durations for Reliability Sensitive Data Type in Scenario 4.2 (10 packets/second)](image)

Figure 4.21: Simulation Durations for Reliability Sensitive Data Type in Scenario 4.2

**General Application Data Type**

Figure 4.22 and 4.23 shows that the average throughputs and packet delivery ratios of a route selected by the delay and the remaining battery aware algorithms are very low. This is because the route, RC_A (the first route selected by both algorithms), contains a malicious node that has dropped many packets, reducing both average throughputs and packet delivery ratios. The route selected by the trust aware algorithm (RC_{B\rightarrow C}) also suffers from low average throughputs and packet delivery ratios. This is because the bandwidth capacity of RC_B (the first route selected by the trust aware algorithm) is too low to support high data rate traffic.
The FRD framework has made the best decision (has chosen RC\textsubscript{C→B}) in terms of average throughputs and packet delivery ratios. This route has enough bandwidth capacity to cope with the data rate of the general application data type. Although some data packets are dropped during transmission along route RC\textsubscript{C} (the first route selected by FRD), the effect
is not significant. The average throughputs of this route are still at the highest level when compared with the routes selected by the delay, trust and battery remaining aware algorithms.

![Figure 4.24: Simulation Durations for General Application Data Type in Scenario 4.2](image)

The simulation durations for the general application data type (Figure 4.24) are similar to the simulation duration of the reliability sensitive data type (Figure 4.21). That is $RC_{A\rightarrow B}$ and $RC_{A\rightarrow C}$ have the longest simulation duration, $RC_{C\rightarrow B}$ has the second longest simulation duration, and $RC_{B\rightarrow C}$ has the least simulation duration. In this case, the routes selected by the delay aware algorithm ($RC_{A\rightarrow B}$) and by the remaining battery aware algorithm ($RC_{A\rightarrow C}$) are the best for the general application data type in terms of simulation duration.

By looking at the overall performances of these different routing algorithms for the general application data type, FRD also makes the best routing decision compared with the decisions by the other algorithms. The route selected by FRD has the best average throughputs and packet delivery ratios and average simulation duration results. The routes selected by the delay, trust and remaining battery aware algorithms have very low average throughputs and packet delivery ratios.
4.6.2.4 Discussion of Simulation Results

The simulation results described above are summarised in Table 4.20. The table shows for each application type, its QoS requirements along with the performance values from each routing decision. For example, for the delay sensitive data type, it needs a high throughput and high reliability route, so the table contains average throughput and packet delivery ratio results from each of the routing decisions for this application type. For the reliability sensitive data type, the table contains packet delivery ratio and simulation duration. For the general application data type, it considers all RRMTs equally, so all the values of the RRMTs, i.e. average throughput, packet delivery ratio, and simulation duration, are listed for this application type in the table. Basically, this table only shows the simulation results that matter to the QoS requirements of the applications concerned.

<table>
<thead>
<tr>
<th>Application Data Types</th>
<th>Routing Algorithms</th>
<th>Best Decisions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DAA</td>
<td>TAA</td>
</tr>
<tr>
<td>Delay sensitive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average throughput (KBytes/Sec)</td>
<td>16KBps</td>
<td>12KBps</td>
</tr>
<tr>
<td>Packet delivery ratio (%)</td>
<td>14%</td>
<td>12%</td>
</tr>
<tr>
<td>Reliability sensitive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Packet delivery ratio (%)</td>
<td>15%</td>
<td>94%</td>
</tr>
<tr>
<td>Simulation duration (Minutes)</td>
<td>28Mins</td>
<td>2Mins</td>
</tr>
<tr>
<td>General application</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average throughput (KBytes/Sec)</td>
<td>8KBps</td>
<td>12KBps</td>
</tr>
<tr>
<td>Packet delivery ratio (%)</td>
<td>15%</td>
<td>23%</td>
</tr>
<tr>
<td>Simulation duration (Minutes)</td>
<td>7Mins</td>
<td>2Mins</td>
</tr>
</tbody>
</table>

| Total Best               | FRD (5 times)    |

Table 4.20: Summary of the simulation results reflecting on the requirements of the application

From this table, the following observations can be made:

- For the delay sensitive data type, the routing decisions made by FRD provide the highest average throughput and packet delivery ratio.
- For the reliability sensitive data type, both FRD and the trust aware algorithm provide the highest packet delivery ratio results. The delay and the remaining battery aware algorithms provide the longest simulation duration.
- For the general application data type, FRD provides the best average throughput and packet delivery ratio, while the delay and the remaining battery aware algorithm have the longest simulation duration.
- FRD has made 5 best decisions in optimising average throughput and packet delivery ratio for the delay sensitive data type and the general application data type.
For the reliability sensitive data type, it can also select a route with the most optimising packet delivery ratio and simulation duration.

From these results, we can say that, in comparison with its peers, FRD performs best, i.e. it can find a route that could best satisfy the QoS requirements of the three applications. It also can be seen that using a single routing metric type to make a routing decision may not satisfy the QoS needs of dissimilar applications or dissimilar application data types. A single metric type can only be used to identify the quality of a route based on that particular perspective. For example, a delay aware algorithm can only be used to find the best route in terms of bandwidth. It does not indicate whether the route is reliable or not. However, if using more than one routing metric types to make a routing decision, several aspects of a route can be considered.

FRD addresses this issue by considering different routing metric types and takes the requirements of an application into account when making a routing decision for the application. The simulation results have shown that, with this multi-metric and application-aware approach, application-level QoS requirements can be better satisfied. It is also worth emphasising that, the use of FRD is not limited to the routing metric types used in our simulation (i.e. bandwidth, trust and remaining battery metric types). The framework is also capable of accommodating more and other routing metric types; this is dependent on the need of the applications and/or technological advancement.

4.7 Integration of the FRD Framework with the Real Network System

In the previous section, we have discussed the FRD framework and how it works. Here, in this section, we discuss how the FRD framework may be integrated into a real world system.

One of the most interesting questions is who would be in the position to select the FRD parameters (i.e. RRMTs, RRMTWs and a MCDM technique) for an application?. There are three possible approaches to this issue: (1) pre-defined by software developers, (2) by users, and (3) pre-defined by software developers and configurable by users.
For the first approach, the software developers need to consider what QoS requirements an application should have and to choose the FRD parameters accordingly, when they develop the application. In this case, users do not need to be aware that the FRD framework is used. A limitation of this approach is that users of the application do not have any say in the selections of the FRD parameters.

The second approach is to let the users to make the selections. Perhaps with the application there is a tool that can guide a user to make the selections. For example, for the Skype application (Voice over IP) shown in Figure 4.25, a user may be prompted to specify higher-level requirements such as a low delay and a high reliability. Then the FRD framework can do the mapping between these higher-level requirements specified by the user and the set of FRD parameters.

![Image of Skype settings showing QoS options]

Figure 4.25: The example of the high-level requirements of the application called Skype

The last approach is to give full control to the users. Software developers may pre-define all the FRD parameters, but users can also configure them manually. To illustrate this idea,
we use the interface of Microsoft Windows 7 Operating System as an example. Figure 4.26 shows what it may look like if the FRD framework is integrated with the OS. Here, the original Internet Protocol Version 4 (IPv4) suite can be replaced with the Internet Protocol Version 4 with FRD (IPv4/FRD). We assume that the IPv4/FRD has all the functionalities of the original IPv4 suite and it is integrated with the FRD framework.

Figure 4.26: The example of IPv4 Protocol Suite with the FRD Framework

In the properties of the IPv4/FRD suite, there should be an option for users to enable or disable the FRD framework. An example of this is shown in Figure 4.27. Users can either enable or disable the FRD framework by selecting this option. If the FRD framework is disabled, this protocol suite will act the same as original IPv4 suite, and none of the FRD framework features will be activated.
Figure 4.27: The example of the FRD Framework Activation

In addition, we can see in Figure 4.27 that there is a List of Applications Supported by the FRD framework (called LAS). In LAS, there can be several applications registered with the FRD framework. If a user has installed a new application which is supported by the FRD framework, the new application will appear in the LAS list. However, there are some cases that an application is not supported by the FRD framework by default (not included in the LAS). This could be because the application was not developed with a pre-set of the
FRD parameters. In this case, users can also add a setting of the new application and set all the settings (i.e. RRMTs, RRMTWs and MCDM technique) manually.

When the FRD framework is enabled and there are applications which are not shown in the LAS list, these applications will use the setting of the "General Application (Default)" in order to be supported by the FRD framework.

As mentioned above, the FRD framework can be used to select a different route for each different application. The setting for each application can be configured or edited. Figure 4.28 shows an example of an MCDM advanced setting for the general application. Here, at the MCDM technique tab, there is a list of the available MCDM techniques which have been installed into the machine. One of these techniques may be selected by users. Users may also set up a default setting that can be recommended by software developers. In addition, if there is a new MCDM technique available, users may also install it.

![Advanced Settings for General Application](image_url)

Figure 4.28: The example of MCDM Technique Settings for General Applications
Figure 4.29: The example of RRMT and RRMTW Settings for General Applications

For the RRMT tab of the advance settings (shown in Figure 4.29), we can see that each application may have its own set of RRMTs and RRMTWs. The weight of RRMT which is not relevant to the application will be set to 0 as mentioned earlier. The total weight here should equal to 1. If users do not set up these values correctly, they should be shown with an error message. With these settings, the value of the parameters such as the MCRD technique, RRMTs and RRMTWs can be passed down to the network layer. A routing decision can then be made using the data as mentioned early on in this chapter.

4.9 Chapter Summary

Multiple application and application data types are increasingly supported in ad hoc network environments. Different application data types will typically impose different QoS requirements in terms of performance, delay, trust and reliability. This chapter has described a solution to demonstrate that user-level QoS requirements can be better
supported by using different route selection criteria to select the best possible route that could best satisfy a given QoS requirement.

The solution, i.e. the FRD framework, supports routing decision making by using multiple metric types (i.e. multi-criteria routing decision making). To accommodate the diversified application-level QoS requirements, multiple routing metric types have been identified and interpreted in the FRD framework design. In addition, the FRD framework integrates two factors into the decision making process. One is a RRMTW which indicates application/user-level preferences for desirable routing metric types. The other is the use of a MCDM technique which is used to select a route among a set of optional routing candidates measured by multiple routing metric types. The FRD framework combines both factors into a single decision making attribute and allows the use of this attribute to govern the selection of a route, which is the most appropriate one for a given set of complex routing criteria. The solution has overcome some weaknesses exhibited by existing single metric routing algorithms used in ad hoc networks. Simulation results have shown that the FRD framework often outperforms the existing algorithm.
Chapter 5

Conclusions and Future Work

5.1 Thesis Conclusions

The focus of this thesis was on achieving the design of a flexible routing decision to support the requirements imposed by a variety of applications in MANETs. This chapter summarises the work presented in this thesis, gives the contributions and discoveries from the research findings, and recommends future work.

The thesis has presented research backgrounds on MANET routing mechanisms and MANET security issues. Chapter 2 gave an overview of routing mechanisms, then reviewed existing routing protocols. Then the chapter also reviewed security issues in MANET routing (i.e. threats and attacks on MANET routing).

The finding of the research background shows that MANET routing is vulnerable to a security attack more than traditional infrastructure networks because of its self-organise manner, limited resources and dynamic topologies. Designing a secure routing protocol against security attacks in MANETs routing can be difficult and challenging as each category of solutions tend to address a certain class of attacks.

Chapter 3 critically identified the security attacks on the ETX protocol. We found out that in the ETX protocol, there was no mechanism to verify the reverse delivery ratio $d_r$ value received is correct or genuine. This provided an opportunity for a malicious node to fabricate and advertise false $d_r$ values to its neighbours.

In order to counter the attack, this thesis proposed the design and simulation study of a novel solution to counter black hole attacks on the ETX protocol. This novel solution, i.e. the SETX protocol, shifted the task of computing link quality metric values onto the initiator of a communication, and had a built-in mechanism to allow the initiator to verify any probe messages returned by the neighbouring nodes, thus effectively reducing the chance for neighbouring nodes to successfully fabricate link quality data by malicious nodes. Simulation results have shown that the SETX protocol provides a much better
performance than its ETX counterpart in malicious environments. The bigger the network size, the bigger the improvement, and these improvements are achieved with very little overhead costs.

Another finding from the reviewing of research background was that choosing the best route alone was not sufficient to optimise QoS provisioning. Different application types typically imposed different QoS requirements in terms of performance, delay, trust and reliability. Typical routing decision making methods (i.e. route selection methods) in existing ad hoc routing protocols were not efficient in supporting a variety of applications. It did not take into account different application’s requirements into routing decisions. Since different applications could generate different data types, they might have different routing requirements. Some routing requirements (e.g. high performance and high security level) might trade-off between with each other. If a routing protocol chooses a secure route for a delay sensitive application, the route selected may have a high delay which may not meet the minimum requirements of the applications. The routing performance might not be efficient for the application. We have pointed out that routing protocols should select a route regarding the requirements of applications.

The question is how we can map an application routing requirement into a routing decision. The first attempt on the solution was a framework that uses a different routing protocol for different types of applications. We categorised multiple groups of applications in the same type, and also select different routing protocols which will be used for the chosen group of applications. For example, the delay sensitive application group will use a distance vector base like AODV [PER97] to select a route. Secure sensitive application may use TRP [XUE04], and the power aware uses a Power-Aware Routing Protocol [SIN98]. Advantages of this approach are (1) each group of applications selects a different route to transmit their data. This can spread network loads on the same route to other routes, (2) a chosen route can satisfy a routing requirement of an application better than using one routing protocol for all types of applications.

However, the framework was not flexible enough in terms of fitting routing requirements of each application into a routing protocol. For example, if an application is both a delay sensitive and secure sensitive type, this application will need to decide which group that they belong. Using this solution does still not solve the problem entirely as a routing decision will not be able to accommodate both of the requirements at the same time.
Chapter 5 introduced the improved version of the framework called Flexible Routing Decision (FRD). The first change is that the framework will use multiple routing metric types instead of using a complete routing protocol. The second change is to use a weighting method on a routing decision. The weight of a routing metric type is a reflection of how much the routing metric type is relevant to the application group. With this approach, each application, depending on its data types and/or application-level requirements, is associated with one or more routing metrics. When a routing decision is required for the traffic generated by the application, we use a MCDM technique to make the routing decision based on the requirements of the application (in terms of weights). The underlying routing decision process will select the most appropriate metric(s) and examine the values of these metrics to select the best route for the application. The solution has overcome some weaknesses exhibited by existing single metric routing decision methods used in ad hoc networks. Simulation results have shown that the FRD framework often outperforms the existing routing decision methods.

5.2 Contributions

The thesis has made the following contributions and discoveries:

Secure ETX

1. The Secure ETX (SETX) protocol is proposed. The ETX protocol is used to find a low loss rate route in ad hoc networks. However, the original design of ETX protocol does not have a security provision to protect a routing decision process against routing metric value modification and forgery attacks. SETX is designed to thwart the black hole attack on the ETX protocol by introducing a secure method to derive a metric value.

Flexible Routing Decision (FRD) framework

In detail, the FRD framework has the following novelty and novel components.

2. It supports the considerations of multiple routing criteria on a single platform.

3. It considers multiple network-layer routing metric types, and application-layer QoS requirements.
4. A technique to map application-level requirements onto network layer routing criteria or metrics. The technique makes use of a MCDM technique to allow users to specify their preferences for a routing metric type desirable for a given application data type. By using this technique, a routing decision can satisfy the QoS requirements of the application data type better than a routing decision made by a traditional routing algorithm.

**Performance Evaluation**

5. To demonstrate the efficiency and efficacy of the SETX protocol and FRD framework, both protocols have been implemented using NS2 simulation package and the performances of the protocols have been evaluated and compared to that of related work.

**5.3 Future Work**

There are multiple security solutions proposed for securing a routing process in ad hoc networks, and each solution has its own pros and cons. In general, there is a trade-off between the level of security provided to the network routing process and the underlying network performance. The higher the security level, the more expensive the security solution, which usually means the more resources mobile nodes would have to consume in a routing process in terms of processing time, battery consumption, etc. This will adversely affect the underlying network performance. On the other hand, if we do not use any security solution to secure a routing process, then the network operation may be disrupted by malicious attacks, and this will in turn lead to network performance degradation. Therefore, there is a fine balance between the security level provided to a routing process and the performance of the underlying network that can be achieved. To minimise any unnecessary resource consumption in a resource-constrained ad hoc network and to improve its routing efficiency while, at the same time, not to compromise its security, we envisage an adaptive approach to security provisioning in ad hoc routing procedures. In other words, if the underlying environment is more secure, or there is a higher level of trust among the mobile nodes in the network, then we would favour the use of a more efficient and less expensive security solution (which is usually less secure). Otherwise, if the underlying
environment is less secure, experiencing a higher level of risks and security threats, or if the trust level among the communication parties are much lower, then we should use a more stringent security solution that is usually more expensive. We believe, with this adaptive approach, we may be able to introduce less overhead costs and routing delays thus improve the underlying network throughputs while keeping the security risk levels under control.

The future works of the SETX and FRD framework are discussed as followed.

**Future works on the SETX**

In SETX, the protocol provided the security protection to the original ETX protocol against a black hole attack. However, the SETX has also introduced another security attack on its probe message mechanism. This issue is still needed to be addressed. So the future work on the SETX could be the implementation of the probe message protection using the keyed hash function technique (Section 3.3.3.2). This technique can be used to increase the level of protection on the probe message but there are some issues which are needed to be worked on. The main issue is how a symmetric key should be distributed. It can be distributed as a group or as pair-wised key.

For the group key sharing method, it can be difficult to decide whether which neighbouring node will be included into a group? We may decide by selecting all neighbouring nodes within the coverage of the initiator node to be the same group. However, what if a mobile node is moving into and out from the coverage to the initiator node. Do we need to discard the current key and issue another one for this new group? Is this mechanism efficient? This work is seriously needed to be considered.

On the other hand, another method of sharing a key is to use a pair-wise key. Each pair of a mobile node has a key for themselves. The problem of this method is how the initiator node can obtain a list of the neighbouring nodes that is located within the coverage of the node itself. This way, the initiator will need to multicast (i.e. specific a list of intended receiver) probe packets instead of broadcasting them. Each probe message is required to encrypt and attached to a probe packet. In the case that the initiator node does not have an accurate list of the receivers (e.g. some nodes may leave the coverage of the initiator node), it would be wasted of resources to encrypted a probe message for the missing
nodes). This is another work that needs to be considered for this pair-wised key sharing method.

**Future works on the FRD framework**

The FRD framework is not ready to apply to a real-life application. There are two major works that are necessary to be done which are assigning RRMTWs and normalising RRMV scales.

**Assigning RRMTWs**

The value of a RRMTW can be very sensitive to the final routing decision. In some cases, if the value of a RRMTW is changed by a little, the routing decision can be changed too. Here, we use one example to illustrate this. This example has the same settings and condition as Scenario 4.2 in Section 4.6.2 except that the RRMTWs used here (shown in Table 5.1) are different from those used in Scenario 4.2.

<table>
<thead>
<tr>
<th>Application Data Types</th>
<th>RRMTWs</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Delay Metric Type (DM)</td>
<td>Trust Metric Type (TM)</td>
</tr>
<tr>
<td>General Application</td>
<td>0.5</td>
<td>0.25</td>
</tr>
</tbody>
</table>

*Table 5.1: The RRMTW Table of the Example Scenario*

When we increase the RRMTW of DM to 0.5 and reduce RRMTWs of TM and RBM to 0.25. The routing decision is changed. For example, using the RAHP technique, the decision table for this general application data type is given below in Table 5.2.

<table>
<thead>
<tr>
<th>Routing Candidates</th>
<th>RRMTs (RRMTW)</th>
<th>RAHPs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RRMTW&lt;sup&gt;DM&lt;/sup&gt;(0.5)</td>
<td>RRMTW&lt;sup&gt;TM&lt;/sup&gt;(0.25)</td>
</tr>
<tr>
<td>RC&lt;sub&gt;A&lt;/sub&gt;</td>
<td>(0.7/0.7) x 0.5</td>
<td>(0.1/1) x 0.25</td>
</tr>
<tr>
<td>RC&lt;sub&gt;B&lt;/sub&gt;</td>
<td>(0.7/3.06) x 0.5</td>
<td>(1/1) x 0.25</td>
</tr>
<tr>
<td>RC&lt;sub&gt;C&lt;/sub&gt;</td>
<td>(0.7/1.72) x 0.5</td>
<td>(0.95/1) x 0.25</td>
</tr>
</tbody>
</table>

*Table 5.2: A Decision Table for General Application Data Type when using RAHP technique*

As can be seen, the first decision is changed from RC<sub>C</sub> to RC<sub>A</sub> by modifying RRTMWs. This means that the final routing decision could be very sensitive to the values of RRMTWs and care must be taken when selecting the values. This issue will need some techniques or mechanism to define an optimised weight for routing metric types for each application.
Normalising RRM Scales

Another future work is that the FRD framework is the need to define a normalised scale for each RRMV when the WSM technique is used. This is because different RRMTs can have very different scales of RRMV. Examples of maximum and minimum values of different RRMTs are shown in Table 5.3.

<table>
<thead>
<tr>
<th>Metric Type</th>
<th>Value Type</th>
<th>Best Value</th>
<th>Worst Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distance Vector (Hop Count)</td>
<td>Minimization</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>Available Bandwidth (e.g. Mbps)</td>
<td>Maximization</td>
<td>54</td>
<td>1</td>
</tr>
<tr>
<td>Delay, Round Trip Time (Seconds)</td>
<td>Minimization</td>
<td>0</td>
<td>255 (max TTL value from [BIN09])</td>
</tr>
<tr>
<td>ETX</td>
<td>Minimization</td>
<td>1</td>
<td>10000</td>
</tr>
<tr>
<td>Packet delivery ratio (%)</td>
<td>Maximization</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>Trust Value</td>
<td>Maximization</td>
<td>1.0</td>
<td>0</td>
</tr>
<tr>
<td>Remaining Battery Level (%)</td>
<td>Maximization</td>
<td>100</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 5.3: List of Metric Types and their Value Ranges used by the FRD Framework

These RRMTs have different scales. As WSM does not use relative values as in the case for WPM, AHP and RAHP, the RRMVs of RRMTs should be normalised into the same scale.

Normalising the RRMV of a RRMT can be difficult. This is because, for doing so, we will need to know the average value of the RRMT. Simply finding a middle value from the min/max values of the RRMV may not be sufficient. For example, assuming the ETX metric type has min/max values of 1 to 10000. By looking at this value, we may think that the average value of the ETX metric is 5000. However, the ETX value of 5000 can be very rare; a route with 1 hop and a forwarding delivery ratio of 0.0002 scores this ETX value. This link condition is not very common. Therefore, scaling the RRMV of a RRMT should use the most common value not a middle value.

Here, we want to convert the original ETX value into a scale between 0 and 100, where 0 is the worst value and 100 is the best value (as shown in Figure 5.1). We assume that the most common ETX value of our network is 10. If Route A (RC_A) has the ETX metric value of 10, then this route would have a metric value in the normalised scale of 50 (which is in the middle of the scale). Another example is Route B (RC_B). It has the ETX value of 15. So this route should have a metric value in the normalised scale below 50. This is
because we know that a route with the ETX value of 15 is worse than a route with the ETX value of 10. For Route C (RC_C), it has the ETX value of 5. This ETX value is better than the common value. So the ETX value of RC_C in the normalised scale should be higher than 50.

Figure 5.1: The example of normalising ETX metric values

These considerations are necessary to make the scoring system of the WSM technique more accurate. The more information we have on the common/average value, the more accurate routing decision the framework can make. However, this future work will need to consider this issue for every single RRMT that we want to use. In conclusion, the aims of this research are to determine the security and performance of routing functions in MANETs, to investigate existing solutions, and to improve the performance of routing functions in terms of security and efficiency. These have been achieved but there remain other possible avenues of future work.
Bibliography


[POO10] Poorter, E.D., AODV Routing protocol, Department of Information Technology (INTEC), Ghent University - IBBT, [Online] Available at: http://idraproject.net/content/aodv-routing-protocol, [Accessed 13 March 2013].


[TSI01] Tsirigos A. and Haas Z. J., Multipath routing in mobile ad hoc networks or how to route in the presence of frequent topology changes, In *Proc. IEEE*


Appendix

A. Probabilities of Guessing $n$ Probe Message in the SETX Protocol

Following the discussion on Section 3.4.2.1, the probabilities that the malicious node $M$ can guess $n$ probe message correctly are discussed in this section. Since a probe message is 64 bits long, the probability that node $M$ will guess 1 probe message correctly is $2^{-64}$. Let's call this number $a$. If $a$ is the probability to guess 1 probe message correctly, the probability that node $M$ will not guess any probe correctly will be $1-a$ (called $b$).

If there are 2 probe messages that node $M$ misses and node $M$ want to guess the missing probe messages, there are 4 events in this case. These are,

1. Node $M$ guesses two probes correctly.
2. Node $M$ guesses the first probe message correctly but it guesses the second probe message wrong.
3. Node $M$ guesses the first probe message wrong but it guesses the second probe message correctly.
4. Node $M$ guesses both probe messages wrong.

For event (1), there are two independent sub-events which are (1.1) Node $M$ guesses the first probe correctly and (1.2) Node $M$ guesses the second probe correctly. As the probability of (1.1) is $a$, and the probability of (1.2) is $a$, the probability of this event is $a^2$. If we use the same idea with event (2), (3) and (4), we will find the probabilities of these four events as in Table A.1.

<table>
<thead>
<tr>
<th>Event</th>
<th>Probability to guess the first probe message</th>
<th>Probability to guess the second probe message</th>
<th>Probability of each event (in terms of $a$ and $b$)</th>
<th>Probability of each event (the actual values)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>$a$</td>
<td>$a$</td>
<td>$a^2$</td>
<td>$2^{-64}$</td>
</tr>
<tr>
<td>(2)</td>
<td>$a$</td>
<td>$b$</td>
<td>$ab$</td>
<td>$(2^{-64})(1 - 2^{-64})$</td>
</tr>
<tr>
<td>(3)</td>
<td>$b$</td>
<td>$a$</td>
<td>$ab$</td>
<td>$(2^{-64})(1 - 2^{-64})$</td>
</tr>
<tr>
<td>(4)</td>
<td>$b$</td>
<td>$b$</td>
<td>$b^2$</td>
<td>$(1 - 2^{-64})^2$</td>
</tr>
</tbody>
</table>

Table A.1: List of all events when Malicious node $M$ guessing 2 probe messages
If node $M$ misses 3 probe messages, then we can have 8 events that node $M$ will guess the probe messages correctly. These are,

1. Node $M$ guesses three probes correctly.
2. Node $M$ guesses the first and the second probe messages correctly but it guesses the third probe message wrong.
3. Node $M$ guesses the first and the third probe messages correctly but it guesses the second probe message wrong.
4. Node $M$ guesses only the first probe message correctly but it guesses the second and the third probe messages wrong.
5. Node $M$ guesses the first probe message wrong but it guesses the second and the third probe messages correctly.
6. Node $M$ guesses the first and the third probe messages wrong but it guesses the second probe message correctly.
7. Node $M$ guesses the first and the second probe messages wrong but it guesses the third probe messages correctly.
8. Node $M$ guesses all probe messages wrong.

Then the probabilities of all events of node $M$ guessing the probe messages are shown in the following table.

<table>
<thead>
<tr>
<th>Event</th>
<th>Probability to guess the first probe message</th>
<th>Probability to guess the second probe message</th>
<th>Probability to guess the third probe message</th>
<th>Probability of each event (in terms of $a$ and $b$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>$a$</td>
<td>$a$</td>
<td>$a$</td>
<td>$a^3$</td>
</tr>
<tr>
<td>(2)</td>
<td>$a$</td>
<td>$a$</td>
<td>$b$</td>
<td>$a^2b$</td>
</tr>
<tr>
<td>(3)</td>
<td>$a$</td>
<td>$b$</td>
<td>$a$</td>
<td>$a^2b$</td>
</tr>
<tr>
<td>(4)</td>
<td>$a$</td>
<td>$b$</td>
<td>$b$</td>
<td>$ab^2$</td>
</tr>
<tr>
<td>(5)</td>
<td>$b$</td>
<td>$a$</td>
<td>$a$</td>
<td>$a^2b$</td>
</tr>
<tr>
<td>(6)</td>
<td>$b$</td>
<td>$a$</td>
<td>$b$</td>
<td>$ab^2$</td>
</tr>
<tr>
<td>(7)</td>
<td>$b$</td>
<td>$b$</td>
<td>$a$</td>
<td>$ab^2$</td>
</tr>
<tr>
<td>(8)</td>
<td>$b$</td>
<td>$b$</td>
<td>$b$</td>
<td>$b^3$</td>
</tr>
</tbody>
</table>

Table A.2: List of all events when Malicious node $M$ guessing 3 probe messages

If there were more number of probe messages missing, it will be more complicated to find a probability. However, there is a probability theory of probability mass function [STE11] that we can apply this problem to find the probability. That is,

$$P(k) = \binom{n}{k} a^k b^{n-k}$$

where $P(k)$ is a probability of node $M$ getting exactly $k$ out of $n$ probe messages correctly.
For example, if we want to know the probability of node \( M \) guessing exactly 1 probe messages correctly (same as we calculate earlier), the probability will be:

\[
P(1) = \binom{3}{1} a^{1} b^{3-1} = \frac{3!}{1!(3-1)!} ab^2 = 3ab^2
\]

This value is the same as we calculate manually from the table. In addition, one may want to know the probability that node \( M \) can guess at least 1 probe message correctly. The event to guess at least 1 probe message correctly can be the event that node \( M \) gets exactly 1 probe message correctly (i.e. \( P(1) \)) or the probability of the event that node \( M \) gets exactly 2 probe messages correctly (i.e. \( P(2) \)) or the probability of the event that node \( M \) gets all the probe messages correctly. That means,

\[
P(\text{at least } 1) = P(1) + P(2) + P(3)
\]

\[
= \left( \binom{3}{1} a^{1} b^{3-1} \right) + \left( \binom{3}{2} a^{2} b^{2-1} \right) + \left( \binom{3}{3} a^{3} b^{3-3} \right)
\]

\[
= 3ab^2 + 3a^2b + a^3
\]

Now, if node \( M \) does not receive any probe message from initiator \( I \) at all (i.e. missing 15 probe messages, and \( n = 15 \)), the probabilities that node \( M \) guesses exactly \( k \) probe messages correctly is:

\[
P(k) = \binom{15}{k} a^{k} b^{15-k}
\]

For example, if \( k = 1 \) or the probability that node \( M \) guesses 1 out of 15 probe messages correctly will be:

\[
P(1) = \binom{15}{1} a^{1} b^{15-1} = \frac{15!}{1!(14)!} ab^{14} = 15ab^{14}
\]
B. NS2 Configurations

B.1 TCL Command for Malicious Nodes Implementations

In our simulation model implementation, a TCL command is used to indicate which node is a malicious node [TAL09]. For example, the following command,

\[
\texttt{\$ns\_ at 0.01 } \left[ \texttt{\$node\_ (0) set ragent\_} \right] \texttt{ malicious}
\]

indicates that mobile node 0 will start to act as a malicious node at time 0.01 second. After the TCL command is connected to the c++ code, the keyword “malicious” will be bound with a variable named as “malicious”, which has been declared at the main program of the routing protocol (aodv-uu.h and aodv-uu.cc). It is processed in the method “command()”.

\[
\begin{align*}
\text{if (argc == 2) } & \{
\text{if (strcmp(argv[1], “malicious”) == 0) } \{
\text{malicious = true;}
\text{return TCL_OK;}
\}\}
\end{align*}
\]

In the above code, the variable “malicious” is an indicator that the node will act as a malicious node. The action of the malicious node should then be modified. As mentioned above, the behaviour of a black hole node is to drop a data packet. Therefore, whenever the malicious node receives a data packet, it simply drops it. In the programming source code of the aodv-uu.cc file, there is a method, “\texttt{recv(Packet *p, Handler *)}”. This method is invoked when a node received a data packet. The following code is added into the method, so that a data packet will be dropped.

\[
\begin{align*}
\text{if ( malicious == true ) } & \{
\text{drop(p, DROP_RTR_ROUTE_LOOP);}
\}\}
\end{align*}
\]
If the variable “malicious” is set to “false”, the data packets received will be processed and be forwarded on.

### B.2 Nominal Radio Range

The nominal radio range of mobile nodes is set to 50 meters throughout the thesis simulation settings. This is an average range of wireless IEEE 802.11b and 802.11g [HOE04]. As in NS2 version 2.26, the default value of the range is set at 250 meters, there is a need to reconfigure the simulation to provide more realistic environment. To customise a nominal radio range of a node in the simulation package, we need to change a value of the variable called “RXThresh_” and “CSThresh_” [NS2-MAN]. These variables are the signal strength reception threshold and the carrier sensing threshold. If the received signal strength is greater than these thresholds, a packet can be successfully received. If we modify this value, then the range of the transmission can be changed. The question is how much this value should be in order to get a transmission range of 50 meters.

Here, a program called “threshold.cc” is used. This program is included in NS2 version 2.26 package under the path “ns-2.26/indep-utils/propagation”. The program calculates the threshold value of the variable “RXThresh_” and “CSThresh_” from the desired range. With output of this program, to set the transmission range of 50 meters, we need to set the value of these variables to 7.69113e-08. Then we can set this value to the variable for our simulation by adding the following command in our tcl script file (where we configure our scenario).

```
Phy/WirelessPhy set RXThresh_ 7.69113e-08
Phy/WirelessPhy set CSThresh_ 7.69113e-08
```

We have tested whether this modification is valid by creating 6 scenarios. Each scenario contains 2 mobile nodes, node A and node B. Both nodes are not moving and they are fix at the designate locations until the end of the simulation. The first scenario locates node A at (0, 0) and node B at (47, 0). There is a 47m distance between the two nodes. The following scenarios have longer distances which are 48m, 49m, 50m, 51m and 52m.

---

1 In a format of (x, y) where x is the position of x axis and y is the position of y axis. This assumes that all
We tested by sending 1000 packets from node A to node B. The results show that node B from the first 4 scenarios, which have the distance between two nodes below or equal 50m, can receive all data packets without any loss. However, the scenarios which have the distance longer than 50m (i.e. 51m and 52m) cannot detect that node B receives a data packet at all. This means the modification in this case should be valid.
C. WPM Algorithm for FRD Decisions

C.1 WPM Technique

The WPM technique uses a pair-wise comparison between routing candidates. It compares the first routing candidate (i.e. $RC_1$) with the second routing candidate (i.e. $RC_2$). The worse routing candidate is dropped. The better one is kept and is then compared with the next routing candidate (i.e. $RC_3$) in the routing table, and so on. This process continues until we have compared all of the available routing candidates. The last remaining routing candidate will be the best route.

For example, there are three routing candidates; $RC_1$, $RC_2$ and $RC_3$. The source node wants to find the best route for application data type A. Application data type A has two RRMTs which are ETX and trust metric types. The $WPMR(RC_1/RC_2)$ will be,

$$WPMR(RC_1/RC_2) = \left( \frac{RRMV_{ETX}^{1}}{RRMV_{ETX}^{2}} \right)^{RRMTW_{ETX}^{ETX}} \times \left( \frac{RRMV_{Trust}^{1}}{RRMV_{Trust}^{2}} \right)^{RRMTW_{Trust}^{ETX}}$$

If the value of $WPMR(RC_1/RC_2)$ is more than 1, then $RC_1$ is better than $RC_2$ and vice versa. If $WPMR(RC_1/RC_2)$ equals to 1 (this means that both $RC_1$ and $RC_2$ are equally as good for the given application-layer requirements). The routing candidate that is registered into the routing table first will be selected. For example, if the source node receives the first routing candidate (i.e. $RC_1$) through the first RREP that arrived before the second routing candidate (i.e. $RC_2$). $RC_1$ is believed to be a better route in terms of less delay (that is why it arrives first), so $RC_1$ will be selected.

The above described process is coded into a $WPM()$ function. To compare any two routing candidates, the function takes the values of three parameters/variables; $RC[x][]$, $RC[y][]$ and $RRMT[A][]$. $RC[x][]$ and $RC[y][]$ denote routes, $RC_x$ and $RC_y$, respectively. These variables contain all RRMs for all routing candidates. For example, $RC[x][]$ contains the RRMs of $RC_x$. $RRMT[A][]$ is an array of RRMTWs of application data type A. This variable contains the RRMTWs of the RRMTs chosen for the

---

1 The first RC is the first routing candidate that the source node registered into its routing table first. The first RC is basically derived from the first arrival RREP.
application data type. If there are other application data types, e.g. application data type B, the RRMTWs contains in the \texttt{RRMT[B][]} will have an array of different values from \texttt{RRMT[A][]}.

For example, the RRMT domain for the node mentioned in example of Figure 4.2 has three RRMTs: ETX, TM and RBM types. So \texttt{RC[x][]} contains \texttt{RRMV^ETX_x}, \texttt{RRMV^TM_x} and \texttt{RRMV^RBM_x}. \texttt{RC[y][]} contains \texttt{RRMV^ETX_y}, \texttt{RRMV^TM_y} and \texttt{RRMV^RBM_y}. \texttt{RRMT[A][]} contains \texttt{RRMTW^ETX_A}, \texttt{RRMTW^TM_A} and \texttt{RRMTW^RBM_A}. The \texttt{WPM()} function returns the result of \texttt{WPMR^A(RC_x/RC_y)} as shown in Equation 4.2. The pseudo code of the \texttt{WPM()} function is as follows.

\begin{verbatim}
WPMR WPM(RC[x][] , RC[y][] , RRMT[A][])
   Result_Sum = 1.0
   for eachRRMT
      Temp = ( RC[x][RRMT] / RC[y][RRMT] )
      Temp = Temp \^{ RRMT[A][RRMT] }
      Result_Sum = Result_Sum \* Temp
   WPMR = Result_Sum
return WPMR
\end{verbatim}

where \texttt{RC[x][RRMT]} denotes the value (i.e. \texttt{RRMV}) of the routing metric type RRMT for route \texttt{RC_x} (i.e. \texttt{RRMV^{RRMT}_x}). "\^" denotes a power operator. \texttt{RRMT[A][RRMT]} is the weight of the corresponding routing metric type, RRMT, of application data type A (\texttt{RRMTW^{RRMT}_A}). \texttt{WPMR} is the ratio between \texttt{RC_x} and \texttt{RC_y}.

The function called \texttt{FRD_WPM()} searches for the best route using the QoS requirements defined by an application data type. This function makes use of the \texttt{WPM()} function to find a better route and then drop a worse route. The function has two parameters/variables which are (1) \texttt{RC[]} which is an array that contains all available routing candidates (e.g. \texttt{RC[1][]}, \texttt{RC[2][]}, \texttt{RC[3][]}, ..., \texttt{RC[n][]}) and (2) \texttt{RRMT[A][]} which is an array containing all \texttt{RRMTWs} of application data type A (i.e. \texttt{RRMTW^{ETX}_A}, \texttt{RRMTW^{Trust}_A} and \texttt{RRMTW^{Battery}_A}). The pseudo code of the \texttt{FRD_WPM()} function is shown as follows:

\begin{verbatim}
bestRC FRD_WPM(RC[][], RRMT[A][])
   best_route_index = 1
   for i from 2 to m step by 1
\end{verbatim}
if WPM(RC[i][],RC[best_route_index][],RRMT[A][])>1 then
  best_route_index = i
bestRC = RC[best_route_index][]
return bestRC

where, \( m \) is the total number of routing candidates. \( best\_route\_index \) is the index pointing to the current best route. This value starts with a value of 1, which points to the first routing candidate, i.e. RC[1][] . The index changes whenever a new and a better routing candidate is found. Once all the routes are compared (i.e. after exit the for loop), RC[best_route_index][] indicates the best route for the given application data type (i.e. application data type A). The variable bestRC is then assigned to the best route, and it will be returned as the routing decision.

C.2 RAHP Technique

As mentioned in Section 4.4.5, the RAHP technique uses the maximum RRMV from all the available routing candidates (i.e. \( \max_{1 \leq i \leq m}(RRMV_i) \)) to find a relative RAHP value. Then this relative value will be used to find a RAHPS. With the RAHP technique, the following steps are involved to find a best route. These are (A) searching for a maximum RRMV of each RRMT, (B) calculating a relative RAHP value for each RRMV of each routing candidate, and (C) calculating RAHPS for each routing candidate by using the relative RAHP values.

(A) Searching for a maximum RRMV of each RRMT

This maximum search algorithm must search from all available routing candidates (i.e. RC[][]) with a given RRMT. The pseudo code of this algorithm is shown below.

```plaintext
MAX_VALUE MAX_OF(RRMT, RC[][])
MAX_VALUE = RC[1][RRMT]
for i from 2 to m step by 1
  if RC[i][RRMT] > MAX_VALUE then
    MAX_VALUE = RC[i][RRMT]
return MAX_VALUE
```
The algorithm searches RRMVs of the RRMT of all the routing candidates (i.e. $RC[i][RRMT]$ from $i = 1$ to $m$, where $m$ is the number of the RRMTs in the RRMT domain). The algorithm sets the $MAX_VALUE$ with the value of the first RRMV (i.e. $RC[1][RRMT]$). This $MAX_VALUE$ is compared with the all the remaining RRMVs one by one. If there is a RRMV which is higher than the value of $MAX_VALUE$. Then the $MAX_VALUE$ is updated with that higher RRMV. After the end of the execution, the $MAX_VALUE$ is the highest RRMV from all routing candidates.

(B) Calculating relative value

After the maximum RRMV has been obtained, a relative RAHP value can be calculated. This is calculated from the RRMV divided by the maximum RRMV and multiply with RRMTW of the same RRMT. For example, to calculate a relative RAHP value of $RC_i$, we calculate $(RC[i][RRMT] / MAX_OF[RRMT]) \times RRMT[A][RRMT]$. Then we can use this relative RAHP value to find RAHPS in step (3).

The algorithm used to calculate a RAHPS is $RAHP()$. This algorithm is actually an implementation of Equation 4.5. The pseudo code of this algorithm is shown below.

```plaintext
RAHPS RAHP(RC[x][] , RRMT[A][])
result_Sum = 0.0
for each RRMT
    temp = ( RC[x][RRMT] / MAX_OF[RRMT])
    temp = temp * RRMT[A][RRMT]
    result_Sum = result_Sum + temp
RAHPS = result_Sum
return RAHPS
```

(C) Calculating a RAHPS for each routing candidate

The RAHPS for each routing candidate will be calculated. A routing candidate with the highest RAHPS will be selected as the best route based on the given criteria. The pseudo code to determine the best route is shown below.

```plaintext
bestRC FRD_RAHP(RC[][], RRMT[A][])
bestRC = RC[1][]
best_RAHPS = RAHP(RC[1][]{}, RRMT[A][]{})
```
for $i$ from 2 to $m$ step by 1

    temp_RAHPS = RAHP(RC[1][]), RRMT[A][])

if temp_RAHPS > best_RAHPS then

    bestRC = RC[i][]
    best_RAHPS = temp_RAHPS

return bestRC

Once the best route has been selected, the source node can begin data packet transmission. The packets will be sent through the selected route.
D. Full Simulation Results for SETX protocols

D.1 Simulation Results of Scenario 3.1

Figure D.1.1: Average Throughputs in Scenario 3.1 with the data rate of 1 packet/second

Figure D.1.2: Average Throughputs in Scenario 3.1 with the data rate of 10 packets/second

Figure D.1.3: Average Throughputs in Scenario 3.1 with the data rate of 100 packets/second

Figure D.1.4: Packet Delivery Ratios in Scenario 3.1 with the data rate of 1 packet/second

Figure D.1.5: Packet Delivery Ratios in Scenario 3.1 with the data rate of 10 packets/second

Figure D.1.6: Packet Delivery Ratios in Scenario 3.1 with the data rate of 100 packets/second
Figure D.1.7: Simulation Durations in Scenario 3.1 with the data rate of 1 packet/second

Figure D.1.8: Simulation Durations in Scenario 3.1 with the data rate of 10 packets/second

Figure D.1.9: Simulation Durations in Scenario 3.1 with the data rate of 100 packets/second

Figure D.1.10: Control Packet Counts in Scenario 3.1 with the data rate of 1 packet/second

Figure D.1.11: Control Packet Counts in Scenario 3.1 with the data rate of 10 packets/second

Figure D.1.12: Control Packet Counts in Scenario 3.1 with the data rate of 100 packets/second
Figure D.1.13: Control Packet Rates in Scenario 3.1 with the data rate of 1 packet/second

Figure D.1.14: Control Packet Rates in Scenario 3.1 with the data rate of 10 packets/second

Figure D.1.15: Control Packet Rates in Scenario 3.1 with the data rate of 100 packets/second
D.2 Simulation Results of Scenario 3.2

Figure D.2.1: Average Throughputs in Scenario 3.2 with the data rate of 1 packet/second

Figure D.2.2: Average Throughputs in Scenario 3.2 with the data rate of 10 packets/second

Figure D.2.3: Average Throughputs in Scenario 3.2 with the data rate of 100 packets/second

Figure D.2.4: Packet Delivery Ratios in Scenario 3.2 with the data rate of 1 packet/second

Figure D.2.5: Packet Delivery Ratios in Scenario 3.2 with the data rate of 10 packets/second

Figure D.2.6: Packet Delivery Ratios in Scenario 3.2 with the data rate of 100 packets/second
Figure D.7: Simulation Durations in Scenario 3.2 with the data rate of 1 packet/second

Figure D.8: Simulation Durations in Scenario 3.2 with the data rate of 10 packets/second

Figure D.9: Simulation Durations in Scenario 3.2 with the data rate of 100 packets/second

Figure D.10: Control Packet Counts in Scenario 3.2 with the data rate of 1 packet/second

Figure D.11: Control Packet Counts in Scenario 3.2 with the data rate of 10 packets/second

Figure D.12: Control Packet Counts in Scenario 3.2 with the data rate of 100 packets/second
Figure D.2.13: Control Packet Rates in Scenario 3.2 with the data rate of 1 packet/second

Figure D.2.14: Control Packet Rates in Scenario 3.2 with the data rate of 10 packets/second

Figure D.2.15: Control Packet Rates in Scenario 3.2 with the data rate of 100 packets/second
D.3 Simulation Results of Scenario 3.3

Figure D.3.1: Average Throughputs in Scenario 3.3 with the data rate of 1 packet/second

Figure D.3.2: Average Throughputs in Scenario 3.3 with the data rate of 10 packets/second

Figure D.3.3: Average Throughputs in Scenario 3.3 with the data rate of 100 packets/second

Figure D.3.4: Packet Delivery Ratios in Scenario 3.3 with the data rate of 1 packet/second

Figure D.3.5: Packet Delivery Ratios in Scenario 3.3 with the data rate of 10 packets/second

Figure D.3.6: Packet Delivery Ratios in Scenario 3.3 with the data rate of 100 packets/second
Figure D.3.7: Simulation Durations in Scenario 3.3 with the data rate of 1 packet/second

Figure D.3.8: Simulation Durations in Scenario 3.3 with the data rate of 10 packets/second

Figure D.3.9: Simulation Durations in Scenario 3.3 with the data rate of 100 packets/second

Figure D.3.10: Control Packet Counts in Scenario 3.3 with the data rate of 1 packet/second

Figure D.3.11: Control Packet Counts in Scenario 3.3 with the data rate of 10 packets/second

Figure D.3.12: Control Packet Counts in Scenario 3.3 with the data rate of 100 packets/second
Figure D.3.13: Control Packet Rates in Scenario 3.3
with the data rate of 1 packet/second

Figure D.3.14: Control Packet Rates in Scenario 3.3
with the data rate of 10 packets/second

Figure D.3.15: Control Packet Rates in Scenario 3.3
with the data rate of 100 packets/second
D.4 Simulation Results of Scenario 3.4

Figure D.4.1: Average Throughputs in Scenario 3.4 with the data rate of 1 packet/second

Figure D.4.2: Average Throughputs in Scenario 3.4 with the data rate of 10 packets/second

Figure D.4.3: Average Throughputs in Scenario 3.4 with the data rate of 100 packets/second

Figure D.4.4: Packet Delivery Ratios in Scenario 3.4 with the data rate of 1 packet/second

Figure D.4.5: Packet Delivery Ratios in Scenario 3.4 with the data rate of 10 packets/second

Figure D.4.6: Packet Delivery Ratios in Scenario 3.4 with the data rate of 100 packets/second
Figure D.4.7: Simulation Durations in Scenario 3.4 with the data rate of 1 packet/second

Figure D.4.8: Simulation Durations in Scenario 3.4 with the data rate of 10 packets/second

Figure D.4.9: Simulation Durations in Scenario 3.4 with the data rate of 100 packets/second

Figure D.4.10: Control Packet Counts in Scenario 3.4 with the data rate of 1 packet/second

Figure D.4.11: Control Packet Counts in Scenario 3.4 with the data rate of 10 packets/second

Figure D.4.12: Control Packet Counts in Scenario 3.4 with the data rate of 100 packets/second
Figure D.4.13: Control Packet Rates in Scenario 3.4 with the data rate of 1 packet/second

Figure D.4.14: Control Packet Rates in Scenario 3.4 with the data rate of 10 packets/second

Figure D.4.15: Control Packet Rates in Scenario 3.4 with the data rate of 100 packets/second
D.5 Simulation Results of Scenario 3.5

Figure D.5.1: Average Throughputs in Scenario 3.5 with the data rate of 1 packet/second

Figure D.5.2: Average Throughputs in Scenario 3.5 with the data rate of 10 packets/second

Figure D.5.3: Average Throughputs in Scenario 3.5 with the data rate of 100 packets/second

Figure D.5.4: Packet Delivery Ratios in Scenario 3.5 with the data rate of 1 packet/second

Figure D.5.5: Packet Delivery Ratios in Scenario 3.5 with the data rate of 10 packets/second

Figure D.5.6: Packet Delivery Ratios in Scenario 3.5 with the data rate of 100 packets/second
Figure D.5.7: Simulation Durations in Scenario 3.5 with the data rate of 1 packet/second

Figure D.5.8: Simulation Durations in Scenario 3.5 with the data rate of 10 packets/second

Figure D.5.9: Simulation Durations in Scenario 3.5 with the data rate of 100 packets/second

Figure D.5.10: Control Packet Counts in Scenario 3.5 with the data rate of 1 packet/second

Figure D.5.11: Control Packet Counts in Scenario 3.5 with the data rate of 10 packets/second

Figure D.5.12: Control Packet Counts in Scenario 3.5 with the data rate of 100 packets/second
Figure D.5.13: Control Packet Rates in Scenario 3.5 with the data rate of 1 packet/second

Figure D.5.14: Control Packet Rates in Scenario 3.5 with the data rate of 10 packets/second

Figure D.5.15: Control Packet Rates in Scenario 3.5 with the data rate of 100 packets/second
E. Full Simulation Results for FRD

E.1 Simulation Results of Scenario 4.1

Figure E.1.1: Average Throughputs in Scenario 4.1 for Reliability Sensitive Data Type (10 packets/second)

Figure E.1.2: Average Throughputs in Scenario 4.1 for General Data Type (50 packets/second)

Figure E.1.3: Average Throughputs in Scenario 4.1 for Delay Sensitive Data Type (100 packets/second)

Figure E.1.4: Packet Delivery Ratios in Scenario 4.1 for Reliability Sensitive Data Type (10 packets/second)

Figure E.1.5: Packet Delivery Ratios in Scenario 4.1 for General Data Type (50 packets/second)

Figure E.1.6: Packet Delivery Ratios in Scenario 4.1 for Delay Sensitive Data Type (100 packets/second)
Figure E.1.7: Route Breaking Times in Scenario 4.1 for Reliability Sensitive Data Type (10 packets/second)

Figure E.1.8: Route Breaking Times in Scenario 4.1 for General Data Type (50 packets/second)

Figure E.1.9: Route Breaking Times in Scenario 4.1 for Delay Sensitive Data Type (100 packets/second)
E.2 Simulation Results of Scenario 4.2

Figure E.2.1: Average Throughputs in Scenario 4.2 for Reliability Sensitive Data Type (10 packets/second)

Figure E.2.2: Average Throughputs in Scenario 4.2 for General Data Type (50 packets/second)

Figure E.2.3: Average Throughputs in Scenario 4.2 for Delay Sensitive Data Type (100 packets/second)

Figure E.2.4: Packet Delivery Ratios in Scenario 4.2 for Reliability Sensitive Data Type (10 packets/second)

Figure E.2.5: Packet Delivery Ratios in Scenario 4.2 for General Data Type (50 packets/second)

Figure E.2.6: Packet Delivery Ratios in Scenario 4.2 for Delay Sensitive Data Type (100 packets/second)
Figure E.2.7: Simulation Durations in Scenario 4.2 for Reliability Sensitive Data Type (10 packets/second)

Figure E.2.8: Simulation Durations in Scenario 4.2 for General Data Type (50 packets/second)

Figure E.2.9: Simulation Durations in Scenario 4.2 for Delay Sensitive Data Type (100 packets/second)