A generic trust based routing protocol against internal attacks in manet

Cai, Ruojun

2016

Cai, R. (2016). A generic trust based routing protocol against internal attacks in manet.
Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/68953>

<https://doi.org/10.32657/10356/68953>

**A GENERIC TRUST BASED ROUTING PROTOCOL
AGAINST INTERNAL ATTACKS IN MANET**

CAI RUOJUN

School of Electrical and Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfilment of the requirement for the degree of
Doctor of Philosophy

2016

Acknowledgements

Before I met my supervisor, Associate Prof. Peter Han Joo Chong, I did not believe I can become a Ph.D student. I feel privileged that I had a chance to study under his supervision. I would also like to express my gratitude to another supervisor, Associate Prof. So Ping Lam. Prof. So had helped me to proofread my thesis patiently and discussed with me about my work. His attitude towards student and work shape my personal growth.

I would like to thank Dr. Xuejun Li for showing me how to write high quality paper hand by hand and taught me some very useful skills about research tools; Dr. Lingfu Xie who is my guide to a lot of issues and always helpful in many ways; Dr. G. G. Md. Nawaz Ali who helped me to revise my manuscripts carefully and patiently; My mid-term examiner Assistant Prof. Rongxin Lu from NTU, who is always very kind to me. Even if I was not a registered student of his class, he still allowed me to attend the class and emailed me those lecture notes personally.

My thanks also go to many other friends and colleagues at Center of Infocomm Technology. Thanks to Cherry Aung for sharing programming knowledge with me. Thanks Kai Wen and Chi Cheng for being a friend in need, with whom I can talk about almost everything.

I would also like to thank Mr. David Koh Koon Wee. He is the first person who brought me into engineering world, sharing a lot of his experience about life, giving me advice when I needed. My thanks also go to Mrs. Kim Lim, who offered me a lot of help when the first day I landed on this foreign country. I even want to thank those private

bosses who had provided me a job at the restaurants, convenient stores and café shops. Although I can only get minimum wages, it was a great help for me at that time. And these experiences changed my attitude towards life. I am not good at expressing my feeling, but those hands offered to me in my most difficult time will never be forgotten.

Two important friends, Chen Li and Xiao Yu, have always been inspiring me in the life. They told me how important it is to choose the road less traveled by if two roads diverged in a wood, and taught me how to act as a lady but think like man. I cherish the precious friendships with Xin Dai, Simin Lv, Chi Chen, Xin Chen and many others. I felt very lucky to have met you and still have you in my life.

Thanks Yancheng Ren for all the joy and sorrow he has brought to me, giving me the dependence and independence I wanted and understanding me. Lastly, I would like to express my gratitude to my parents for their consistent support and encouragement during my growth. Even in times of disagreement, my parents have always respected my decision and supported me in a loving manner.

Cai Ruojun

Nov. 2015

Table of Contents

Acknowledgements.....	i
Table of Contents.....	iii
Abstract.....	vii
List of Abbreviations	ix
List of Figures	xi
List of Tables	xiv
Chapter 1. Introduction	1
1.1 Motivation.....	2
1.2 Objective.....	3
1.3 A Commonly Used Routing Protocol: DSR.....	4
1.3.1 DSR Route Discovery.....	4
1.3.2 DSR Route Maintenance	6
1.3.3 Additional Features.....	7
1.4 Contribution.....	8
1.5 Organization.....	10
Chapter 2. Literature Review	11
2.1 Implementation and Goal of Attacks in DSR	12
2.1.1 Misuse RREQ Message in DSR	13
2.1.2 Misuse RREP Message in DSR.....	18
2.1.3 Summary of Malicious Actions	22
2.2 Conventional Reliable Routing Scheme	22
2.2.1 Secure Ad hoc On-demand Distance Vector	23

2.2.2	Ariadne.....	24
2.2.3	Deficiencies of Conventional Secured Routing Scheme	25
2.3	General Framework of Trust-Based Routing.....	27
2.3.1	Trust Formulation	28
2.3.2	Trust Dissemination.....	36
2.3.3	Decision Making.....	37
2.4	Open Issues and Directions.....	38
2.5	Conclusion	39
Chapter 3. Self-checking Trust Routing Scheme against Active Black Hole Attacks		43
3.1	Introduction.....	44
3.1.1	Acknowledgement (ACK) Based Scheme.....	45
3.1.2	Next Hop Indication Scheme	46
3.1.3	Bait Detection Scheme.....	46
3.1.4	Monitoring Based Scheme.....	48
3.1.5	Multipath Based Scheme	49
3.2	Self-checking Trust Routing Scheme	50
3.2.1	Updating/Maintaining Neighborhood Topology	51
3.2.2	Liar Checking	53
3.2.3	Consistency Checking.....	54
3.3	Security against internal attacks	57
3.3.1	Internal Attacker Type A	57
3.3.2	Internal Attacker Type B	60
3.4	Enhanced Self-checking Trust Routing Scheme	61
3.5	Simulation Model and Results.....	64

3.5.1	Attacker Pattern	65
3.5.2	Performance Metrics	66
3.5.3	Baseline and Benchmark	67
3.5.4	Simulation Results	68
3.6	Conclusion	78
Chapter 4. An Evolutionary Self-Cooperative Trust Scheme against Passive Attacks		80
4.1	Introduction.....	81
4.2	Evolutionary Self-Cooperative Trust Scheme (ETS)	84
4.2.1	Self-Detection	86
4.2.2	Cooperative Detection	89
4.3	Simulation and Results	95
4.3.1	Attack Patterns	96
4.3.2	Performance Metrics	97
4.3.3	Baseline and Benchmark	98
4.3.4	Simulation Results and Analysis	99
4.4	Conclusion	112
Chapter 5. Simulation Study of Attacks Under Network Coding Environment.....		114
5.1	Introduction.....	115
5.2	Intra-flow Network Coding	116
5.3	Security Threats in Network Coding System	120
5.3.1	Forwarder Set Selection.....	120
5.3.2	Batch Generation	121
5.3.3	Acknowledgement Replication.....	123
5.4	Simulation Results	124

5.4.1	Simulator and Simulation Parameters.....	125
5.4.2	Attack Patterns	126
5.4.3	Performance Metrics and Result Analysis.....	126
5.5	Challenges of Secured NC System Design.....	131
5.6	Conclusion	133
Chapter 6. Conclusion and Future Work		135
6.1	Conclusion	135
6.2	Recommendation for Future Work	139
Author's Publications		141
Bibliography		143

Abstract

Mobile Ad Hoc Networks (MANETs) have wide applications in practice, such as rescue, emergency operations, and battlefield communications. MANETs consist of a group of mobile nodes that dynamically exchange data among themselves without the reliance on any fixed infrastructure or centralized administration. Nevertheless, mobility and self-organizing characteristics of MANETs cause the change of topology in an unpredictable way. Due to the limited transmission range, each node has to seek assistance of its neighboring nodes to help relay the data to transmit to other nodes. The reliable neighboring nodes are important for successful data transmission from a source node to a destination node.

The early works providing reliable routing mainly rely on cryptographic tools to prevent malicious nodes from injecting false information into network. However, these reliable routing schemes are always coupled with prerequisites, such as authentication center, key management mechanism, which are not feasible in MANETs. Furthermore, cryptographic operations are considered computationally expensive on resource constrained mobile nodes. The most important issue is that these schemes are incapable of tracking internal attackers who are authenticated to participate in the network but misbehaved during the routing process, e.g. black hole and grey-hole attackers. Thus, there is a need to look for a lightweight approach to provide a reliable ad hoc routing.

In this thesis, we present two trust-based routing schemes to prevent different routing disruption attacks, such as active black hole, passive black hole, and grey-hole nodes. The most attractive feature is that our proposed schemes are not only capable of

preventing those well-known routing disruption attacks, but also immune to collusion attack. Even the internal attackers might know how our proposed trust-based security mechanism works; they still cannot attack the system successfully. In the first scheme, we propose a neighborhood connectivity based trust scheme targeting on the active black hole attacks. In this scheme, every node is required to exchange neighbor information before route discovery, and then uses collected neighbor information to verify each received route reply message. By utilizing previously collected neighbor information, we can greatly increase the robustness of the designed system against internal attackers. In the second scheme, a trust-based routing scheme that emulates human cognitive process is proposed. The mobile nodes will exchange trust-level values and evaluate these values based on their own cognitive judgment. Eventually, each node can dynamically evolve its cognition to exclude malicious entities, such as passive black holes and selfish nodes. The distributed nature of our design allows both of the proposed schemes to be easily implemented on any existing routing protocols to handle routing disruption attacks. We have integrated our proposed trust-based routing schemes into dynamic source routing (DSR) protocol and studied their performances under different adversarial network environments. Simulation results affirm that our proposed trust-based routing schemes promote network scalability and ensure the effective routing under the presence of various routing attackers in MANETs.

List of Abbreviations

AODV	Ad-hoc On-demand Distance-Vector Routing
AF	Active Forge
ACK	Acknowledgement
CABH	Colluded Active Black Hole
CTS	Clear to Send
DSR	Dynamic Source Routing
DR	Drop
DoS	Denial of Service
ETS	Evolutionary Trust Scheme
ESCTS	Enhanced Self-checking Trust Scheme
FR	Forge Reply
GH	Grey Hole
MANETs	Mobile Ad Hoc Networks
MF	Modify and Forward
MABH	Mixed Active Black Hole
MRD	Mixed Routing Disruption
MC	Malicious Collective
NTT	Neighborhood Topology Table
NJR	Neighbor Judgment Record
NC	Network Coding
PBH	Passive Black Hole

PDR	Packet Delivery Ratio
RREQ	Route Request
RREP	Route Reply
RERR	Route Error
RTS	Request to Send
RHC	Receiving History Cache
RD	Route Disruption
RI	Route Invasion
RC	Resource Consumption
SCTS	Self-Checking Trust Scheme
SABH	Single Active Black Hole
SHC	Sending History Cache
SAR	Self-Analysis Record
TTL	Time to Live
TIR	Trust Information Record

List of Figures

Figure 1.1 Example of route discovery.....	5
Figure 1.2 Example of maintenance.	6
Figure 2.1 RREQ_MF_RI.....	16
Figure 2.2 RREP_FR_RD.....	20
Figure 2.3 RREP_FR_RI.	21
Figure 2.4 Unified framework of trust process.....	28
Figure 2.5 An example of network topology.....	31
Figure 2.6 Neighbor recommendation trust model.	37
Figure 2.7 Complete framework of trust routing system.....	41
Figure 3.1 Passive versus active black hole attack.	45
Figure 3.2 Example of bait detection scheme.....	48
Figure 3.3 Updating/Maintaining neighborhood-topology.....	52
Figure 3.4 Liar checking procedure.....	54
Figure 3.5 Consistency checking.	56
Figure 3.6 Region with maximum number of nodes.	59
Figure 3.7 Average number of neighbors.	60
Figure 3.8 A simple network topology.	61
Figure 3.9 Colluded active black hole.	62
Figure 3.10 Percentage of packet loss due to faked path.....	64
Figure 3.11 Test 1-Number of attacker versus PDR.....	69
Figure 3.12 Test 1-Number of attackers versus throughput.	69
Figure 3.13 Test 1-Number of attackers versus routing overhead.....	70

Figure 3.14 Test 1-Number of attackers versus delay.	71
Figure 3.15 Test 2-Maximum speed versus PDR.	72
Figure 3.16 Test 3-ESCTS against internal attacker type A.	73
Figure 3.17 Test 3-ESCTS against internal attacker type B.	74
Figure 3.18 Test 3-ESCTS versus varying internal attacker.....	74
Figure 3.19 Test 5-Performance evaluation under lossy environment	76
Figure 3.20 Test 6-Packet delivery ratio comparison.	77
Figure 3.21 Test 6-Routing overhead comparison.....	78
Figure 4.1 Self-detection procedure.....	89
Figure 4.2 Cooperative detection procedure.....	94
Figure 4.3 Test 1-Number of attacker versus PDR.....	100
Figure 4.4 Test 1-Number of attacker versus throughput.	100
Figure 4.5 Test 1-Number of attacker versus routing overhead.	101
Figure 4.6 Test 1-Number of attacker versus delay.....	102
Figure 4.7 Test 2-Maximum speed versus PDR.	103
Figure 4.8 Test 2-Maximum speed versus cache overflow	103
Figure 4.9 Test 3-Simulation area versus PDR.....	105
Figure 4.10 Test 3-Simulation area versus hops.....	105
Figure 4.11 Test 4-Simulation time versus PDR under (a) GH and (b) PBH.....	107
Figure 4.12 Test 5-Robustness test of ETS on (a) GH and (b) PBH.	108
Figure 4.13 Test 6-Accuracy rate of cooperative detection under (a) GH and (b) PBH.	110
Figure 4.14 Test 6-Link quality versus PDR	111
Figure 4.15 Test 7-Comparison with TDSR.....	112
Figure 5.1 Intra-flow NC based routing in wireless ad hoc networks.	117

Figure 5.2 Structure of existing intra-flow systems and the security vulnerabilities.....	124
Figure 5.3 Effect of varying number of attackers on throughput.	128
Figure 5.4 Effect of varying number of attackers on end-to-end delay.	128
Figure 5.5 Effect of varying number of flows on throughput.	129
Figure 5.6 Effect of the size of simulation area on throughput.	130

List of Tables

Table 2.1: Possible modification fields in RREQ.....	14
Table 2.2: Possible modification fields in RREP.	18
Table 2.3: Summary of malicious action.	22
Table 2.4: Comparison of trust-based routing schemes.....	41
Table 3.1: Hello message format.	51
Table 3.2 Neighborhood topology table.	51
Table 3.3 Extended Hello message format.	63
Table 3.4 Simulation parameters.	64
Table 3.5 Test4-PDR versus varying threshold values.....	75
Table 3.6 Specific simulation parameters in test 6.	77
Table 4.1 SHC of node Nx	88
Table 4.2 RHC of node Nx	88
Table 4.3 Neighbor judgment record (NJR).	90
Table 4.4 Utility matrix of node Nx	90
Table 4.5 Summary of notations for cooperative detection.....	91
Table 4.6 Simulation parameters.	95
Table 5.1 Summary of notations for batch generation.....	119
Table 5.2 Simulation parameters.	125

Chapter 1

Introduction

The proliferation of mobile devices has led to the growth of mobile ad hoc networks (MANETs). MANETs consist of a group of mobile nodes that dynamically exchange data among themselves without the reliance on any centralized administration or fixed base station. Self-organizing characteristic enables MANETs to be easily established in a wide variety of disparate situations, such as rescue, emergency operations, and battlefield communications etc. However, mobility and self-organizing characteristics of MANETs cause the change of topology in an unpredictable way. Therefore, most of the time, each mobile node with limited transmission range has to seek assistance of its neighboring nodes for data transmission. As a result, the performance of MANETs largely depends on the reliable routing among nodes.

During the last decade, extensive studies have been conducted on the routing manners in MANETs, resulting in several matured routing protocols [1-4]. However, all these routing protocols are designed with an assumption that all nodes are fully trusted and willing to cooperate with each other. Thus, they are vulnerable to malicious nodes that are not cooperative or disobey the routing regulations. In addition, the absence of fixed infrastructure, the nature of open transmission media and the dynamically changing topology, all these features make the design of reliable and secure routing protocols in MANETs becomes more challenging.

1.1 Motivation

Several secure routing protocols have been proposed [5-9] to defense against various routing attacks. The security of such schemes relies on various cryptographic tools to prevent routing attackers from injecting false information into the network, which in turn secures the routing path. They utilize digital signature to prevent attacker from masquerading other's identity, symmetric key to provide confidentiality of some fields of the routing message and hash function to prevent malicious modification of the packet header. However, a careful review of these secure routing protocols reveals that most of them presume the existence of a centralized or distributed third party in the network. This assumption is also coupled with pre-existing shared secret keys between nodes inside the network. Unfortunately, these assumptions are invalid in MANETs. As we know, MANETs have the diverse nature of being impromptu, thus they inherently oppose the dependence upon prerequisites. Furthermore, cryptographic operations, such as computing digital signature and verification, are considered computationally expensive on resource constrained mobile nodes.

Most importantly, all these secure routing schemes only focus on preventing external attacker, but incapable of tracking internal attackers who are authenticated to participate in the network but misbehaved during the routing process, such as black hole attackers and grey-hole attackers. Therefore, it is necessary to look for a lightweight and practical defense mechanism to ensure reliable ad hoc routing under various adversary network environments launched by those internal attackers.

1.2 Objective

In order to reduce computational overhead and sustain the improvised nature of MANETs, trust concept derived from social science has been borrowed as a new approach to address the security issues raised by internal attacks in MANET routing. Generally speaking, trust-based defense mechanism is a distributed prevention technique designed under assumption that every node inside the network knows how the trust-based routing scheme works. Each node will dynamically evaluate the trust levels of other nodes based on their own experiences and store this information in their databases. Each node may also disseminate their derived trust-level information to other interested parties to help others to build up their trust information databases. Finally, each node can utilize the trust-level information stored in its database to make various decisions, such as routing path selection, misbehaving node isolation, and relationship establishment. In other words, this defense mechanism can be considered as an add-on security component on top of the existing ad hoc routing protocols to ensure reliable routing.

In this work, we would like to develop some pragmatic trust-based routing schemes and verify their performance based on dynamic source routing (DSR) protocol. These trust-based routing schemes are not only provide a lightweight approach to identify those common routing disruption attackers, but also immune to compromised nodes. In other words, as internal attackers are compromised, they may know how our trust-based defense mechanism works. However, even the internal attackers cannot get much advantage to bring negative impact to the network. In addition to the individual attacks, our proposed trust-based routing schemes can also eliminate collusion attacks that are launched by the colluded parties.

1.3 A Commonly Used Routing Protocol: DSR

Since our proposed trust-based routing schemes are designed based on dynamic source routing (DSR) protocol, we will first briefly introduce this routing protocol in this section.

Dynamic Source Routing (DSR) is a typical and commonly used routing protocol by MANET working groups [2]. As nodes in the network are mobile and wireless transmission conditions keep changing such as interference between nodes, all routing has to be automatically determined and maintained by DSR executed on each participated network node. Using DSR, the network can remain completely self-organizing, without requiring any existing infrastructure or administration. All nodes cooperate to forward packets for each other to allow communication over multiple hops between nodes not directly within wireless transmission range of one another. The core functions of DSR protocol can be divided into two mechanisms, which are *route discovery* and *route maintenance*. These two mechanisms operate entirely *on-demand*. For example, source node only launches route discovery process when it has data packets to transmit, but does not have a route to its destination. Intermediate node executes route maintenance when it realizes the link between itself and next hop in current routing path is broken. Route discovery and route maintenance mechanisms are not only improve the network connectivity, but also guarantee the flexibility of MANET operation.

1.3.1 DSR Route Discovery

Assume that if any given node **S** originates a data packet destined for a target node **D**, it places a source route giving the sequence of nodes that the packet should follow in the header of the packet. Normally, node **S** obtains a suitable source route by searching

its *Route Cache* of routes previously learned, but if no route is found in its cache, it initiates the route discovery phase to find a new route to target **D** dynamically. Figure 1.1 illustrates this process. To initiate the route discovery, **S** transmits a *route request* (RREQ) message as a single local broadcast packet, which is received by all one-hop neighbors of node **S**. Each RREQ message identifies the *source* and *destination* of this route discovery in IP header, and also contains a unique *request ID* determined by the source node **S** of this RREQ. Each RREQ message also contains a record listing the address of each intermediate node through which this RREQ message has been forwarded. Initially, this route record is an empty list created by the source of the route discovery. When neighboring node, such as node **A**, receives this RREQ, and it did not receive this RREQ before by checking the *source IP* and *request ID* of this request message, it appends its address to the route record of RREQ and keeps broadcasting until this request being receive by the destination. Otherwise, node **A** will drop this RREQ message to avoid repeated broadcasting. When destination receives this RREQ, it returns a *route reply* (RREP) message that is routed back to the source, together with a copy of the accumulated routes extracted from RREQ packet. Finally, the source will cache all replied routes in its *route cache* and choose the shortest route for data packets transmission.

Figure 1.1 Example of route discovery.

1.3.2 DSR Route Maintenance

In DSR each intermediate node that transmitting the packet is responsible to ensure the packet has been received by the next hop along the source route. The packet is retransmitted (up to a maximum number of attempts) until this confirmation of receipt is received, such as acknowledgement from the MAC layer. For example, as show in Figure 1.2, node **S** originates a data packet to node **D** along source route that passes through nodes **A**, **B**, **C** and **D**. When node **B** realizes that the link between itself to node **C** is broken, it generates a *route error* (RERR) message back to the source, stating that the link between **B** and **C** is not connected. All nodes that receive this error message will remove any route using this link from their route caches. For source node **S** sending retransmission or other data packets to this same destination **D**, if node **S** has another route to node **D**, it can send the packet immediately through the new route. Otherwise, node **S** may perform a new route discovery process for the destination **D**.

Both route discovery and route maintenance phase in DSR can function over unidirectional link, but when DSR is used on top of MAC protocol such as IEEE 802.11, these processes can take advantage of additional optimization, such as reply route request (RREQ) message in the reverse path. This is because IEEE 802.11 itself requires bi-directional exchange of RTS and CTS messages in the protocol.

Figure 1.2 Example of maintenance.

1.3.3 Additional Features

DSR makes aggressive use of route cache and source route. Since in DSR the source route always contains a complete path, no additional mechanism is needed to prevent routing loop. As long as the same IP address does not appear twice in the source route, which means this routing path is loop free. Nevertheless, several additional optimizations have been proposed by the authors of the protocol to further improving the performance, as described in the followings:

- *Caching overheard routing information:* A node forwarding or otherwise overhearing any packet may add the routing information from that packet to its own route cache for future use. For example the source route in a data packet, the route being returned in a RREP or the accumulated route record in a RREQ may all be cached. This feature can help nodes to learn routing information.
- *Automatic route shorting:* When a node overhears a packet with source route, it checks unused portion of this source route. If this node is not the intended next node for the packet, while its IP address appears in the later unused portion of the source route, this node will send a *gratuitous route reply* append with a better path to the source. For example in Figure 1.1, node **C** overhears the data packet being transmitted from **A** to **B** for later forwarding to itself, node **C** will return a *gratuitous reply* with a shorter route to the source of this packet. In this example, the route in *gratuitous reply* will be the sequence of hops S-A-C-D.
- *Replying request use cached route:* When an intermediate node receives a RREQ message for which it is not the requested destination, it checks whether it has a route to the requested destination in its route cache. If it finds, it generates a route reply, sets the route record to the sequence of nodes that the RREQ has been traversed to it

and concatenated with the path from itself to the requested destination that are stored in its route cache. Through this way, DSR reduces flooding of request message and saves network bandwidth to some extent.

- *Packet salvaging:* When a node sends a route error message, it attempts to salvage the data packet that triggered this error message. This node will try to find an alternate route from its route cache that can bypass the failed link to salvage this data packet. However, if no alternate route can be found, salvaging is failed.

1.4 Contribution

The contributions of this thesis are summarized as follows:

- We propose a self-checking trust routing scheme that requires all nodes to use previously collected neighbor information to verify each received route reply message before keeping forwarding the message back to the source node. By utilizing previously collected neighboring information, we are not only able to identify the traditional active black hole attacks, but also can greatly increase the robustness of our scheme in front of internal attackers. This self-checking trust routing scheme is executed by every node independently, and it does not rely on any extra hardware or computational intensive cryptographic tools. Thus, it can be easily applied on existing routing protocols with minor modification, which makes it suitable for practical implementation in MANETs.
- We develop an evolutionary self-cooperative trust scheme to defend against passive routing attacks. This proposed defense mechanism that emulates human cognition can promote network scalability and ensure reliable routing in MANETs via three aspects. Firstly, self-detection mechanism utilizes mobility to perform trust analysis.

This mechanism provides a baseline that ensures the accuracy of trust evaluation. Secondly, cooperative detection can further improve the performance and enhance the robustness of this trust system. On the one hand, cooperative detection speeds up the trust evaluation process and mitigates trust information shortage issue. On the other hand, voting concept implied in the cooperative detection ensures that as long as the majority of the nodes inside the network are honest, forming a malicious collective will not increase the trust value of malicious peers enough in order for them to have impact on the network. Thirdly, evolutionary operation of our scheme enables nodes to explore diverse form of networks, to alleviate the impact of false information provided by the dishonest entities and correct their cognition by themselves.

- Since network coding has emerged as a promising paradigm for designing the MANETs routing, we conduct a thorough study of security aspects of intra-flow network coding based routing system. Firstly, we extract the common components that can provide an overall picture of the most existing intra-flow network coding based routing protocols. Secondly, we analyze the security vulnerabilities of each extracted common component that composes the entire network coding system. Lastly, we simulate various attackers' behaviors under different network environments to evaluate the impact that the attackers may bring to the network. This study provides an insightful round up of all possible security threats throughout each operation of network coding based wireless routing systems, highlighting open questions and the road ahead. It serves as a preliminary guide for the future design of a secured network coding based routing protocol.

1.5 Organization

The report is organized as follows. Chapter 1 provides some background information about MANETs and our work. Chapter 2 provides an overview of trust-based routing design under adversarial MANETs environment. In this chapter, we firstly adopt a systematic approach to explore all possible misbehaved actions that may degrade the performance of DSR. After that, we present a general framework that can help us to classify and understand most of the existing trust-based routing schemes. Chapter 3 proposes a self-checking trust scheme to prevent active black hole attacks. Chapter 4 presents an evolutionary self-cooperative trust scheme to prevent from various passive routing disruption attacks. Since network coding has emerged as a new paradigm for designing wireless routing protocols in recent years, we also conduct a detailed analysis of various routing attacks under network coding environment in Chapter 5. Finally, we conclude our works and highlight some future research directions in Chapter 6.

Chapter 2

Literature Review

In order to design an effective prevention mechanism that can achieve reliable routing and immune to various routing attacks, it is necessary to understand all possible ways that an attacker may misbehave during the routing process and the consequences of its misbehaviors. Classification of routing attacks has been discussed in several literatures [10, 11]. They mainly express the types of attack like black hole, routing loop, wormhole, denial of service, but without a thorough study on how the attackers launch these attacks and their attacking goals. Inspired by author *P. Ning et al* [12], our analysis of various routing attacks consists of two dimensions. The first dimension is a set of possible ways that an attacker may take to misuse a routing message. The second dimension is a set of goals that the attacker wants to achieve. We extend this analysis approach to Dynamic Source Routing (DSR) protocol, and not to the Ad-hoc On-Demand Distance Vector (AODV) as addressed in [12]. After a thorough study of various possible misbehaviors executed by the attackers, we can easily realize the deficiencies of those conventional cryptographic prevention schemes and understand the necessity of the trust-based routing protocol. Despite a couple of surveys on trust management [13-15], a comprehensive review of trust-based routing protocols in MANETs is lacking. Therefore, in this chapter we also review the design issues of those existing trust-based routing protocols. After going through this chapter, we can get a better understanding of the following questions:

- 1) How malicious nodes can achieve their attacking goals by disobeying the routing regulations?
- 2) What kind of defense technique is capable of alleviating the impact caused by malicious attackers?
- 3) Which kind of defense mechanism may be vulnerable to those compromised nodes that know how the defense mechanism works?
- 4) How these defense mechanisms can be integrated to establish a secure and practical trust-based routing protocol?

By answering these questions, we actually get a preliminary guide to the design of trust-based routing protocol.

2.1 Implementation and Goal of Attacks in DSR

As mentioned earlier, in this section we thoroughly study the misbehaviors of attacker pertaining to DSR. Firstly, we list out all possible ways that a malicious node may take to implement routing attacks. After that, we identify the attacking goals from attacker's perspective. All the possible malicious actions that can be executed by the attacker during route discovery process in DSR are divided into the following categories:

- *Drop (DR)*: The attacker simply drops the routing messages.
- *Modify and Forward (MF)*: When receiving a routing message, the attacker modifies one or several fields in the message and then forwards the message to its neighbors via unicast or broadcast.
- *Forge Reply (FR)*: The attacker sends a forged RREP message in response to a received RREQ message.

- *Active Forge (AF)*: The attacker sends a faked RREP without being triggered by receipt of any RREQ message. Similarly, the attacker can also originate a faked RREQ message without the intention to discover any routing path.

The second dimension to analyze routing attacks is a set of goals that attackers want to achieve. Although several survey papers listed out plenty types of routing attacks [10, 16], they can be classified into following goals:

- *Route Disruption (RD)* is to break down an existing route or prevent a new route from being established [17].
- *Route Invasion (RI)* attempts to add an attacker into routing path between two communicating nodes. Once an attacker achieves this goal, it can drop or sniff any passing-by data packets without being noticed [18, 19].
- *Resource consumption (RC)* is to consume the constrained resources like bandwidth, storage space, battery power at other individual nodes [20].

In the rest of this section, we will examine all mentioned malicious actions that are executed by the attacker on the route discovery messages of DSR. For simplicity, we use the form, *Message Type_Malicious Action_Goal*, to express a malicious action that an attacker executes on a particular routing message to achieve the specified goal. E.g. RREQ_MF_RD means the attacker can achieve route disruption (RD) by modifying and forwarding (MF) a RREQ message.

2.1.1 Misuse RREQ Message in DSR

Drop RREQ Message (RREQ_DR)

This type of action refers to simply dropping the received RREQ message. If an attacker dumps received RREQ messages which it is supposed to help broadcast, but

expects others' help to transmit its own packets, this type of attacker is named as *selfish node*. These nodes always want to get help from others without offering to help. It is obvious that through RREQ_DR attacker can only prevent a new route from being established through itself. If the attacker is the only gateway between two parts of the network, it causes network partition. In a dense network, this action has minimum impact, since there are other nodes will help broadcasting the request.

Modify and Forward RREQ (RREQ_MF)

This action refers to attackers modify one or several fields in a received RREQ message before broadcasting it. Table 2.1 lists the RREQ message fields that an attacker may modify, together with the possible modifications. In addition, *source* and *destination* addresses in IP header are also crucial. *Source address* indicates the originator of this RREQ and *destination address* reveals the target of this request. An attacker can create a faked RREQ originating from or targeting to any node by filling the IP address of that node into specified address field. By modifying and forwarding a received RREQ message, attacker can easily achieve route disruption or resource consumption. If network is designed under bidirectional environment, attackers can even achieve route invasion through this simple action. We use examples to elaborate these misbehaviors in the following discussion.

Table 2.1: Possible modification fields in RREQ.

RREQ message field	Possible Modification
Option Type	Change the message type
Request ID	Increase or decrease it to make the message acceptable or unacceptable
Target Address	Replace with other IP address
Address [1...n]	Delete/insert/replace IP addresses

RREQ_MF_RD: An attacker can change *Option Type* field to an undefined message type or change *Request ID* field to a staled number. When neighboring nodes receive this modified RREQ, they have no choice but to drop it. This is because the receivers do not know how to process this message with an undefined message type, and the staled request ID makes the receivers believe that they had processed this RREQ message before. Unlike dropping RREQ by the attacker itself (e.g. Selfish node), in this scenario the benevolent nodes are not able to provide service upon receiving the RREQ message due to the attacker's malicious modification action. This is also regarded as a denial of service (DoS) attack. Since the attacker can prevent a new route from being established to some extent, we claim that the attacker can achieve route disruption through modifying and forwarding a RREQ message.

RREQ_MF_RC: An attacker can simply replace destination address with a non-existent IP address, and this modified RREQ message will be flooding the entire network until time-to-live (TTL) field in IP header becomes zero. Thus, the attacker causes lots of energy wastage.

RREQ_MF_RI: As shown in Figure 2.1, when an attacker (e.g. node **A**) receives a RREQ asking a route to destination **D** from the source node **S**, it can achieve route invasion by broadcasting a modified RREQ message as follows:

- Change the source address (in IP header) to node **D**'s address
- Change target address to a non-existent IP address
- Erase the record in address field and fill a faked path such as **D-A**

When node **D** receives this modified RREQ, it will just drop it. This is because the source address contained in the IP header indicates that this RREQ message is originated from node **D** itself. However, if the network is designed under bidirectional environment,

upon receiving this modified RREQ message, node **S** will update its route cache to node **D** as **S-A-D**. Even when a real reply message comes back, such as **S-2-3-D**, node **S** will still choose the shorter one, which is **S-A-D**, for data transmission.

Figure 2.1 RREQ_MF_RI.

Active Forged RREQ (RREQ_AF)

In contrast to the previously described malicious actions that are triggered when an attacker receives an incoming RREQ message, this type of action refers to an attacker actively broadcasting a faked RREQ message without receiving any RREQ message or the intention to discover a routing path. As long as the network is bidirectional, attacker can achieve route invasion and route disruption by actively broadcasting a faked RREQ message.

RREQ_AF_RI: We use the same topology as shown in Figure 2.1 to elaborate how an attacker achieves route invasion by actively broadcasting a faked RREQ message. For example, when attacker **A** overhears source node is transmitting data packets along the route **S-2-3-D**, node **A** can break the current routing path and achieve route invasion by broadcasting a faked RREQ as follows:

- Set the source address (in IP header) to node **D**'s address
- Set target address to a non-existent node address
- Fill in a faked route, such as route **D-A**, into address field of this RREQ message

When node **S** receives this faked RREQ, it will shift the routing path to **S-A-D** for data transmission, since this faked path is shorter than the current routing path **S-2-3-D**. However, if attacker **A** is not in the transmission range of node **S**, the probability for the attacker to achieve the goal will decrease. For instance, only node **2** can receive this faked RREQ. Even if node **2** updates this routing path, **2-A-D**, to its route cache, it will not inform the source node **S**. This is because the new routing path, **S-2-A-D**, is not shorter than the current routing path **S-2-3-D**.

RREQ_AF_RD: As long as the attacker can achieve route invasion by broadcasting a forged RREQ message actively, it can achieve the route disruption easily. For example, node **S** is transmitting data packets along the route **S-2-3-D** as shown in Figure 2.1. Attacker (Node **A**) can achieve route disruption as follows:

- Set the source address (in IP header) to node **D**'s address
- Set target address to a non-existent node address
- Set address field as **D-4** (node **4** is a non-existent node)

When node **S** receives this forged RREQ message, it will update route cache to destination **D** as **S-4-D**. Once this path is selected for data transmission, it will cause packet drop due to the non-existent node. Benefitting from route maintenance mechanism, once node **S** realizes the link is not in existence through MAC layer acknowledgement, it will repair this link by changing to another routing path or initiating a new route discovery process.

RREQ_AF_RC: If the network environment is unidirectional, the attacker can only achieve resource consumption goal by actively broadcasting a faked RREQ message with a non-existent target IP address. This action will cause the RREQ flooding until the time-to-live (TTL) field in the IP header becomes zero.

2.1.2 Misuse RREP Message in DSR

Drop RREP Message (RREP_DR)

If an attacker is in a strategic location that all RREPs have to pass through, it can easily prevent the route from being established by dropping all passing-by reply messages. However, if there is another RREP message that bypasses the attacker back to the source, the impact of this action is not severe. It is obvious that the attacker can only achieve route disruption by dropping the RREP messages.

Modify and Forward RREP Message (RREP_MF)

RREP_MF refers to an attacker that does malicious modification in certain field of a received RREP message. After that, it forwards this modified RREP back to the source node. Table 2.2 lists the RREP message fields that may be modified by an attacker. In addition, the *source address* and *destination address* in IP header are also very crucial. Source address in the IP header indicates the node generated this reply and the destination address is the node requested for this reply message. By filling a faked IP address in source or destination field in IP header, the attacker can generate a RREP message that is originated from or destined to any node.

Table 2.2: Possible modification fields in RREP.

RREP message field	Possible Modification
Option Type	Change the message type
Address [1...n]	Delete/insert/replace some IP addresses

RREP_MF_RI: If there is only one path between source and destination passing through the attacker, the attacker does not need to modify any field of the received RREP message to achieve route invasion. However, in real situation, the attacker does not know whether there is any other bypassing reply message back to the source. In order to

suppress other possible routing paths, the attacker deletes some IP addresses in the address field. After this modification, this faked reply path contained the attacker may become shorter than other available ones. Therefore, most probably this faked path will be selected for data transmission by the source node. A typical attack that may use this way to achieve route invasion is known as black hole attack.

RREP_MF_RD: Similar to what we have mentioned in *RREQ_MF_RD*, the attacker can easily modify the *Option Type* in a received RREP message to an undefined message type before forwarding it to the next hop node. When the next hop node receives this modified message, it does not know how to process this undefined message but to drop it. Because of the malicious modification, benevolent nodes are not able to provide service for the received RREP message. This is regarded as a denial of service (DoS) attack. However, as long as there is one reply path bypassing the attacker, the impact of this malicious action is not severe.

Forge RREP Message (RREP_FR)

RREP_FR refers to an attacker generating a faked route reply message in response to a received route request (RREQ) message, which is different from modifying a received route reply message (*RREP_MF*) mentioned earlier on. In the rest of the paragraphs, we will use examples to elaborate how the malicious node can achieve route invasion and route disruption by taking this action.

RREP_FR_RD: After receiving a RREQ message, the attacker immediately forges a RREP message going back to the source node, even if it does not know the path to the requested destination. As shown in Figure 2.2, when attacker **A** receives a RREQ message that requires a route to destination **D** from source node **S**, it can generate a faked RREP to achieve route disruption in the following manner:

- Set the source address (in IP header) to a non-existent node (e.g. Node 4) address
- Set the destination (in IP header) to node S's address that is waiting for a reply
- Set the address field in this RREP to **S-1-4-D**

After the attacker forwards this forged reply back to the source, node S will update its route cache according to this faked reply. Since the faked path, **S-1-4-D**, is shorter than the real one **S-1-3-5-D**, this faked path will be selected for data transmission. Once this happens, transmitted data packets will be lost due to the non-existent node 4 in the routing path. Fortunately, after a short period, node 1 may realize the broken link from itself to node 4 through MAC layer acknowledgement and repair it by executing route maintenance procedure.

Figure 2.2 RREP_FR_RD.

RREP_FR_RI: In order to achieve route invasion, an attacker has to convince the source node that it has the shortest path to reach the requested destination. Forging a faked route reply is the most direct way for the attacker to achieve this goal. As shown in Figure 2.3, when an attacker A receives a RREQ message asking for a route to destination D, it immediately generates a faked RREP back to the source as follows:

- Set the source address (in IP header) to its own address (node A's address)
- Set the destination (in IP header) to node S's address that is waiting for a reply
- Set the address field in this RREP to **S-A-D**

Since this faked route is the shortest, eventually the source node will choose this faked path for data transmission. We notice that if the attacker is in the transmission range of

node **S**, it certainly can achieve route invasion. If the attacker is not the direct neighbor of the source node, it can still take this malicious action. However, the probability that the attacker is able to achieve route invasion will decrease. For example, in Figure 2.3, if node **3** does the same thing as attacker **A**, then the faked route contained in the RREP is **S-4-1-3-D**. However, this faked reply route is even longer than the real route **S-4-2-D**, node **3** has no advantage to achieve route invasion.

Figure 2.3 RREP_FR_RI.

Active Forge RREP Message (RREP_AF)

In order to make aggressive use of source routing and route cache, when a node overhears a packet not addressed to it, it will check whether the packet could be routed via itself for a shorter route to be created. If so, the node sends a RREP to the source with this new and better route [2]. Overhearing helps a node to learn routes without having to directly participate in the routing process, but this feature also brings more risks under adversarial environment.

RREP_AF_RD: When attacker (Node A) overhears a data transmission along the route **S-1-3-5-D** as shown in Figure 2.2, it can achieve route disruption by doing exactly the same thing as we mentioned in **RREP_FR_RD**.

RREP_AF_RI: As long as the attacker is a direct neighbor of the source node, it can easily achieve route invasion in the same manner as we described in **RREP_FR_RI**.

2.1.3 Summary of Malicious Actions

In Table 2.3, we summarize all possible actions that may be taken by malicious node on DSR. Under each attack goal we also list some typical attacks. Each cell in this table is marked “Yes” or “No”. “Yes” means attacker can achieve the goal by executing the indicated malicious action. “No” means the opposite. We do not explain the detail about the misusing on route error messages, since the attacker can only achieve route disruption in DSR protocol by misusing the error messages.

When we design a defense mechanism, we have to be aware of all possible malicious actions that may be executed by the attacker. Since we do not know what action the attacker may execute exactly in the next second. In this section, we present a thorough study from attacker’s perspective, which is essential for the design of a reliable routing scheme under adversarial network environment.

Table 2.3: Summary of malicious action.

Goal Action	Route Disruption (dropping packet, DoS)	Route Invasion (black hole)	Resource consumption (flooding, routing loop)
RREQ_DR	Yes	No	No
RREQ_MF	Yes	Yes	Yes
RREQ_AF	Yes	Yes	Yes
RREP_DR	Yes	No	No
RREP_MF	Yes	Yes	No
RREP_FR	Yes	Yes	No
RREP_AF	Yes	Yes	No

2.2 Conventional Reliable Routing Scheme

Secure routing plays an important role in against malicious actions mentioned in the previous section, several proposals have attempted to provide a reliable routing protocol for MANETs. As we will see, conventional reliable routing schemes rely on various

cryptographic tools to prevent routing attackers from injecting false information into the network [5-9]. They utilize digital signature to prevent attacker from masquerading other's identity, symmetric key to provide confidentiality of some fields of the routing message and hash function to prevent malicious modification.

2.2.1 Secure Ad hoc On-demand Distance Vector

Secure Ad hoc On-demand Distance Vector (SAODV) [7] is a proposal for security extension to AODV routing protocol. Before introducing this security mechanism, we briefly elaborate how the original AODV works. Ad hoc on-demand distance vector routing protocol (AODV) [1] shares the on-demand characteristic of DSR, in which a source node discovers route on an as-needed basis via a similar route discovery process introduced in Section 1.3.1. However, in AODV every node uses traditional routing table with one entry per destination to store the most fresh and optimal routing information. This is in contrast to DSR that stores multiple route entries for each destination in the route cache. In the absence of source routing, AODV relies on the routing table to propagate the route reply (RREP) message back to the source and, subsequently, to route data packets to the destination. Therefore, in AODV, each intermediate node decides the next hop to the destination separately based on their routing tables. Since AODV utilizes distance vector principle, it assigns monotonically increasing sequence numbers to routes. This sequence number defines route freshness, as well as hop-count, which defines route optimality.

Secure Ad hoc On-demand Distance Vector (SAODV) [7] utilizes digital signature to authenticate the non-mutable fields of a route request (RREQ) and route reply (RREP) messages. In addition, a new one-way hash chain was created to prevent malicious

altering of the hop-count field, which is the only mutable field in RREQ and RREP message. Because of the end-to-end authentication of non-mutable fields in each route discovery message, and hop-by-hop verification of the hash function, SAODV can effectively prevent route disruption, route invasion and node isolation, which are implemented by modifying and forwarding the route request or route reply messages. In addition, digital signature can prevent an attacker from impersonating a requested destination to generate a faked route reply. Thus, as long as the attackers did not get the private key of the destination, they can neither achieve route invasion nor route disruption by forging a RREP message.

However, SAODV is not able to prevent compromised nodes with valid public and private key pair from executing malicious actions. For example, the compromised node can actively generate a RREQ message with a non-existent destination address and sign it with its public key. This RREQ message with non-existent destination address will flood the network and cause lots of network resources being wasted. Besides, compromised nodes can drop any received packets silently without being caught by SAODV.

2.2.2 Ariadne

Ariadne [6] is a secure reactive routing scheme designed for dynamic source routing (DSR) protocol. In Ariadne, authentication is achieved by using shared secret key between end-to-end nodes and intermediate nodes authentication called TESLA protocol. The design of Ariadne consists of two steps: 1) Ariadne presents a mechanism that allows a destination node to verify the authenticity of the route request (RREQ) message. 2) Ariadne employs a per-hop hashing technique named TESLA broadcast authentication

protocol to verify that no node is missing from the node list in RREQ. For example, when source node broadcasts a RREQ message, it has to include a message authentication code (MAC) computed with a shared secret key K_{SD} over unique data, such as timestamp, source IP address. The destination node can verify the authenticity and freshness of received RREQ message by using shared secret key K_{SD} . Besides this authentication process, the destination node also needs to verify each node in the address field of RREQ, so that it will return a route reply (RREP) along the path that only contains legitimate nodes. In Ariadne, node at each hop authenticates the new information filled into address field of the RREQ by its current TESLA key. When the destination receives the RREQ, it buffers the RREP until the intermediate nodes can release the corresponding TESLA keys. Once the tesla security condition is verified, the destination node will include a MAC in the RREP message and send this reply back to the source. Through a serial complicated authentication, it is not possible for an attacker to achieve route disruption or route invasion by modifying or forging a route discovery message.

Generally speaking, Ariadne provides good defense against modification, fabrication and impersonation through message authentication and verification of routing path. But similar to SAODV, this scheme cannot prevent any malicious packet drop and resource consumption attack executed by the compromised nodes.

2.2.3 Deficiencies of Conventional Secured Routing Scheme

Although we only briefly introduce two conventional secured routing schemes in this section, many of the early proposed secured routing schemes [8, 9, 21] that utilize digital signature to ensure non-repudiation of a communication session, symmetric key to

guarantee end-to-end user authentication, and hash function to secure data integrity have similar requirements and deficiencies. Cryptographic techniques make great contribution on the secured routing with respect to malicious modification and masquerading executed by unauthorized external attackers. However, these conventional secured routing schemes are incapable of tracking any malicious action that is taken by the compromised nodes. For example, one of the most important issues is that they cannot identify compromised nodes that simply drop the packets. In addition, compromised nodes can also actively generate bogus RREP with previous authenticated signature to achieve route invasion.

Besides the above mentioned deficiencies, operation requirements of those hard security approaches (e.g., encryption or authentication techniques) also hinder their successful employments. These are stated as follows: 1) Most of the early secured routing schemes presume the existence of a centralized or distributed third party in the network. This assumption is also coupled with pre-existing shared secret keys between nodes inside the network. However, these assumptions are invalid in MANETs. As we know, MANETs have the diverse nature of being impromptu, thus they inherently oppose the dependence on prerequisites [22]; 2) In MANETs, it is difficult to employ key management due to high computation and communication overhead as well as network dynamic that could require frequent key reassignments [23]; 3) Conventional secured routing schemes relied on hard security components are one time binary type solution. In these schemes, the security checking is either pass or fail. However, the reliability/trustworthiness of received information, quality of information assessment and provision of various levels of access control cannot be done effectively through hard security.

Conventional secured routing schemes tend to be static. There is a need to give a dynamic approach to address the security issues in a dynamic network environment [24].

2.3 General Framework of Trust-Based Routing

In order to reduce computational overhead and sustain the improvised nature of MANETs, trust concept derived from social science has been borrowed as a new approach to address the security issues in MANET routing. Quite a few trust-based routing protocols have been proposed and simulated under various adversarial MANET environments [17, 18, 25]. In the rest of this section, we present a general framework that gives an overall picture of the trust-based routing schemes. Based on this framework, most of the existing trust-based routing schemes can be decomposed, analyzed and compared in a horizontal manner. At the same time, this general framework can help us draw insight into trust-based routing design.

Most of the existing trust-based routing schemes consist of three mechanisms, namely *trust formulation*, *trust dissemination* and *decision making*, as shown in Figure 2.4. The big circles represent the three mechanisms of the trust routing system, with each mechanism consisting of many individual mobile nodes acting as agents. In other words, these three mechanisms are embedded in each individual node. Every node inside the network should understand the whole process of trust analysis. The objective of trust-based routing is either to detect various attackers or to ensure the final selected routing path is secure. To accomplish these objectives, every node has to collect trust or distrust evidences of others. These evidences are the input for the trust analysis. Based on available trust evidence, each node can formulate and calculate the trust-level values of others. After that, some trust-based routing schemes may require nodes to disseminate

those derived trust-level values to other interested peers. If trust dissemination is required, most of the time, those disseminated values will become a new type of evidences that contribute to the trust formulation process. Therefore, trust formulation and dissemination are interacting with each other frequently. Finally, based on those quantified trust values of the other nodes, decision making mechanism can either isolate those nodes with low trust-level values directly or choose the most secure routing path.

Figure 2.4 Unified framework of trust process.

2.3.1 Trust Formulation

In social science, trust is defined as a subjective willingness of one person to rely on the action of another person to achieve certain outcome [26]. However, there is no clear consensus on the definition of trust in a distributed computer network. The first step of trust-based routing design is to transform trust definition from social science into a usable metric according to the available evidences. Therefore, trust formulation is a core mechanism that is responsible for transforming input evidences into a usable trust value through either explicit mathematic equations or an implicitly described algorithm. In the following paragraphs, we will look into trust formulation and further discuss three

important components in it: 1) source of information, 2) information type and 3) trust metric.

2.3.1.1 Source of Information

Source of information is the most important component of trust formulation. It refers to the methods of collecting evidences that are used as the input for trust analysis. The evidences can be gathered from different channels, such as *required feedback*, *direct observation*, *indirect observation*, and *policy-based examination*, based on the requirement of the design.

Required feedback relies on agreed acknowledgement (ACK) to verify the delivery status of packets. In HADOF scheme [27], source node requires the destination to report the number of received packets along current routing path periodically. Once the source node realizes any packet loss from the destination's feedback, it will investigate all intermediate nodes in its current routing path. The investigation results that reveal how many packets an intermediate node had forwarded or dropped are regarded as evidences for trust-level analysis. However, since MANETs are open networks that all nodes should know how the network is configured, it can be assumed that the compromised nodes also know how the trust system works. Therefore, compromised nodes can easily attack the trust system by generating a faked report or modifying the legal feedback from the destination. Most of the time, when the trust system relies on required feedback to collect evidences for trust analysis [28-30], the systems still need to use cryptography to ensure the integrity of the feedback message.

Direct observation is a widely used method to obtain trust evidences in many trust-based routing schemes [31-33]. Most of the early works that use direct observation to

collect trust analysis evidences only take into account the measure of the forwarding behavior of direct neighbors [32, 33]. However, such a measure is not only inadequate for trust-level computation, but also vulnerable to deception. For example, an attacker can modify the packet content before forwarding it, and the attacker can still get a positive rating from trust-level analysis system. To solve this problem, Pirzada et al. proposed a context-aware observation scheme in [31]. In this scheme, each node not only monitors the packet forwarding events, but also verifies the integrity of forwarded packets. For example, as shown in Figure 2.5, when node **A** transmits a packet, such as route request, route reply, route error or data message, it buffers that packet for a specified time interval and monitors the behavior of neighboring node **B** via overhearing. If node **B** forwards the packet properly without any malicious modification in the specified time interval, the successful count C_s of that packet is increased. Otherwise, the failure count C_f will be increased. After node **A** collects these counting numbers, which are the trust evidences, it can compute the trust level of node **B**. With this improvement, direct observation is not only capable of detecting packet dropping but also effective in dealing with malicious modification action to some extent. However, this evidence collection method is not resistance to collusion attack. For example, we assume node **B** and **C** are colluding together in Figure 2.5. When node **A** transmits a packet to node **B**, node **B** can always forwards the packet to node **C** and asks node **C** to drop the packet silently. Through this way, even if node **A** is monitoring the behavior of node **B**, it cannot detect any abnormality and may treat node **B** as a trusted node. In addition, we can notice that the malicious behavior such as actively forging a route reply (RREP) is still unsolved.

Figure 2.5 An example of network topology.

Indirect observation refers to a node gathering the evidences of non-neighboring nodes through intermediate nodes [34-37]. As shown in Figure 2.5, node **B** monitors the behaviors of node **C** and passes the monitoring results to node **A**, this passed information is regarded as the indirect observation evidence gathered by node **A**. On the one hand, indirect observation can help nodes to overcome evidences shortage issue at initial state and speed up trust analysis process. On the other hand, this evidence collection method may make the trust system vulnerable to false accusation or false praise executed by the compromised node. Therefore, we claim that indirect observation method is a double-edged sword. If direct observation that is operated by each individual node can collect enough information to satisfy the requirement of trust-based routing system, it is the best solution. However, if it is inevitable to use indirect observation due to lack of information or any other reason, we have to take care of false accusation and false praise attacks. This is because those compromised nodes may know how the trust system works and attack it accordingly.

Policy-based examination is actually an intrusion detection component. This component can also help nodes collect evidences when they cooperate with each other to detect malicious actions [38-41]. In [38], any intermediate node generating a route reply (RREP) has to indicate the next hop identity to the destination. Upon receiving this RREP from an intermediate node, the source node will send a further request via a disjoint path to the indicated next hop node and ask whether this indicated next hop node is really connected with the required destination. The source node will not trust the

RREP message generated by any intermediate node until it receives a positive reply from that indicated next hop node. The authors had improved the examination process to make it capable of resisting collusion attack in [39]. However, once the compromised attackers know the evidence collection process, they can pretend they are only in charge of forwarding a route reply generated by the destination without indicating any required information in the reply message. Then, the whole system may collapse. Most of the policy-based examination schemes endeavor to detect black hole attackers that attempt to achieve route invasion. However, few of them have addressed compromised attackers who know how the whole defense system works.

As aforementioned, source of information is the most critical component in trust formulation, which determines the baseline of evaluation accuracy. It has to be carefully designed based on the network environment and requirements. For instance, if wireless links are not bidirectional, direct observation though overhearing is not viable. If there are several malicious actions that need to be tracked, we may need to combine different methods to collect trust evaluation evidences. Carefully gathering trust evaluation evidences is the first step to ensure the trust system itself is not easy to be subverted, attacked or compromised.

2.3.1.2 Information Type

Information type is another component in trust formulation that describes whether the trust evidences include positive event, negative event or both, whether the trust evidences are represented as discrete or continuous values. This component influences the application for which a given trust system is most effective as well as determines the types of attacks that are relevant to the system. In CONFIDANT [32], each node only

detects packet drop attacks through direct observation, which is the negative event. In other words, this CONFIDANT system is mainly focused on detecting malicious behaviors. If a system considers both positive and negative events during trust evidence collection process, such as [29], this trust-based routing system is not only trying to identify attackers but also stimulating the cooperation between nodes. In order to avoid slandering, trust evidences collection through indirect observation in [42] only considers positive event. So the trust system is immune to attacks where compromised nodes try to hinder trust evaluation by falsely degrade other's trust levels. Trust system that only considers one type of information can limit the possible attacks on the system itself, but the trust evidences may not be comprehensive. Although representing the trust-level in a discrete or continuous value may not affect the function of the system, using continuous value may be more convenient. This is because most of the formulation results are in real number.

2.3.1.3 Trust Metric

Trust metric actually reveals the physical meaning of trust that is adopted in a designed trust-based routing scheme. Unless the trust definition is clear in the system, it is difficult to transfer input evidences into a usable trust metric. Nevertheless, trust is a very abstract concept that comes from social science, and there is no clear consensus on the definition of trust in a distributed computer network. As a result, the choice of trust metric may also be different for each trust-based routing scheme. There are three widely used trust metrics that measure trust-level by *counting-policy*, *probability* or *uncertainty*.

In the early time and some policy-based examination trust systems, they use counting numbers to indicate the trustworthiness of a node [32, 33, 41]. For example, if node

X notices a malicious action taken by node **Y** during evidence collection process, node **X** will increase the untrusted count of node **Y** by one. Once this untrusted count reaches a predefined threshold, node **X** will regard node **Y** as a completely untrustworthy entity. In [19], the designed system not only counts the number of positive events pertaining to a node, but also records the total number of events interacted with the node. Then the ratio between these two numbers is regarded as the trust-level of the evaluated node. Using counting-policy to indicate the trust-level is the simplest way to map trust evidence and trust metric. These trust-based systems actually define trust as a relatively subjective intuition that emulates the simplest human being's trust behavior. In [34, 43, 44], trust is defined as the likelihood in which an agent will perform certain action before the action is actually done. Thus, these systems represent the trustworthiness of a node by a probability value based on well-established probability distribution. Bayesian approach is the most common model that uses historical evidences to estimate a probability indicating whether a node will behave well in the next round. Nowadays, some researchers also consider the function of uncertainty when designing trust-based routing system [45, 46]. If a subject fully trusts an agent to perform a certain action, then there is no uncertainty. However, if some malicious behaviors of the agent have been caught by the subject, the uncertainty of good action of that agent increases. In other words, the trust level of the agent to the subject is decreased.

In most of the proposed trust-based systems, due to the lack of centralized authority and the reliance on multiple agents, compromised nodes may take advantage of these features to subvert the trust evaluation. It is very important for us to skillfully combine those evidences that are obtained from different channels to enhance the accuracy and robustness of the trust evaluation process. For example, in [45] the authors incorporate

the concept of uncertainty into trust calculation. They use the trust space definition from [47] and extend it by providing a mapping from the evidence space to the trust space as follows:

$$c(p, n) = \frac{1}{2} \int_0^1 \left| \frac{x^p (1-x)^n}{\int_0^1 x^p (1-x)^n dx} - 1 \right| dx \quad (2.1)$$

$$T_1 = \left\{ P_k = c(p, n) \frac{p+1}{p+n+2}, P_n = c(p, n) \frac{n+1}{p+n+2}, u = 1 - P_k - P_n \right\} \quad (2.2)$$

where $c(p, n)$ is the confidence probability that the evidences are certain. x is the probability that a piece of evidence is correct. p and n represent the number of positive and negative events that prove the agent is trustworthy and untrustworthy respectively. After that, the final trust, T_1 , is defined in a 3-tuple space as shown in equation (2.2). P_k , P_n and u represent the belief, disbelief and uncertainty of a subject about an agent respectively. Some researchers use Bayesian probability to model the final trust, T_2 , as shown in equation (2.3) [44, 48, 49]. Although p and n in equation (2.3) have the same meaning as we explained in equation (2.2), the final representation form of trust is different. This difference makes the newcomer be treated differently in two trust models that are represented in equation (2.2) and (2.3), respectively.

$$T_2 = \frac{p+1}{p+n+2} \quad (2.3)$$

For example, when a new node $\mathbf{X}$ joins the network, the other nodes do not have any evidence to evaluate its trust level yet. If a subject uses equation (2.3) to evaluate the trust level of node $\mathbf{X}$, it will be $(0+1)/(0+0+2)=0.5$, which is a neutral view. However, if the subject uses equation (2.2) to compute the trust level of node $\mathbf{X}$, the trust value of node $\mathbf{X}$ is $T_1 = \{P_k = 0, P_n = 0, u = 1\}$, which means not trustworthy. In other words, the final trust computed by equation (2.2) not only reveals the trust-level, but also takes

into account the reason behind such result. This reason may allow the system to give the newcomers a chance to build up their reputation. Although the trust information provide by equation (2.2) seems more comprehensive, the confidence probability of a piece of evidence may not easy to be obtained in real application. Therefore, we conclude that the proper design of trust metric has to be aware of accuracy of trust evaluation, enhance the robustness of the system and implementation in real situation.

2.3.2 Trust Dissemination

Most of the proposed trust-based routing schemes require the trust dissemination mechanism to spread formulated trust-level results to other interested peers. This mechanism can speed up the propagation of trust evaluation results and mitigates the input evidence shortage issue. Although trust dissemination and formulation often interact with each other, it is helpful to separate them for analysis purpose. There are three main approaches to disseminate trust evaluation results: 1) Disseminating trust evaluation results to the requesting peers. 2) Disseminating trust evaluation results to direct neighbors. 3) Disseminating trust evaluation results to the next-hop node on the routing path. For example, in order to detect two colluded malicious nodes, B. Yang [34] proposed a neighbor recommendation trust model. As shown in Figure 2.6, node $\mathbf{M}_1$ and $\mathbf{M}_2$ are colluding, and node $\mathbf{S}$ is transmitting the packets along the route $\mathbf{S-B-M}_1\text{-}\mathbf{M}_2\text{-D}$. According to the neighbor recommendation trust model in [34], even node $\mathbf{M}_1$ behaves well in front of node $\mathbf{B}$, node $\mathbf{B}$ will not increase its trust on $\mathbf{M}_1$ immediately. Instead, node $\mathbf{B}$ will send a trust request (TREQ) to get neighboring nodes' opinion about node $\mathbf{M}_2$, which is the next hop neighbor of $\mathbf{M}_1$. When neighboring nodes $\mathbf{C}$ and $\mathbf{E}$ receive this request message, they will generate a trust reply (TREP) to report their direct

observation results about M_2 . Unless node **B** receives some positive replies from neighbors, it will not increase its trust on node M_1 . This is a typical example that trust evaluation results are only disseminated to the requesting peer through unicast. Most existing trust-based routing schemes use the second dissemination approach that broadcasts trust evaluation results to all its direct neighbors [35, 36, 42]. If the formulated trust value is only disseminated to the next hop node on the path, this implies a routing path from the source to a destination has already been established [19, 50]. Although in some proposed trust-based routing schemes [33, 51], each node only stores trust values of other nodes that it has direct interactions with previously, after some time, the node may get most of the trust information of other peers inside the network. Since nodes are mobile in MANET, it is possible for a node to have direct interaction with any other node inside the network as time elapses.

Figure 2.6 Neighbor recommendation trust model [34].

2.3.3 Decision Making

The last stage of trust-based routing scheme design is the decision making mechanism. Every node uses the obtained trust evaluation results to distinguish those benevolent and malicious peers to achieve reliable routing eventually.

Same as the original routing protocols, this decision making falls on either individual node (e.g. AODV) or source node (e.g. DSR). Decision making falling on individual node means that each node excludes untrusted node by itself based on computed trust-level values [25, 35]. Usually these systems will set a threshold level to distinguish trusted and untrusted nodes. Source node making the decision is similar to the case of source routing. Instead of choosing the shortest path as in the original DSR protocol, source node will choose the path with the highest trust value in trust-based routing scheme. Due to the concern of asymmetric of trust property, in [52], when the source node needs to choose a route for data transmission, it calculates bidirectional trust value of every available routing path and then selects the one with the highest value. However, for simplicity, most of the existing trust-based routing schemes are still using unidirectional trust values at the decision making phase [36, 50].

2.4 Open Issues and Directions

There is a clear trend that sooner or later intelligence will be embedded in each node with cognitive functionality. Researchers defines a cognitive network as having a cognitive process that is capable of perceiving current network conditions and then planning, deciding and acting on those conditions. By adapting to continuously changing network environments, cognitive network is able to improve scalability (e.g., reducing complexity), survivability (e.g., increasing reliability), and QoS (e.g., facilitating cooperation among nodes) as a proactive mechanism. Since trust is an important component that constitutes the cognition system, we believe that trust establishment should also adapt to changing network behaviors, such as attacker behaviors, degree of

hostility and node disconnection due to various reasons. Cognition is not only adaptation, but also a process of learning and reasoning.

Another potential and interesting research direction is to construct a composite trust metric that takes social network properties, human psychology and quality-of-service (QoS) into consideration. Since trust concept comes from social networks, it can be helpful to take human factors into account for establishing trust relationship in the technological networks. Some researchers in this field have been moving in this direction. For example Cho et al. chose intimacy, honesty (social network properties), energy and cooperativeness (QoS properties) for trust evaluation [53]. The best combination for constructing the composite trust metric is still an open question, especially when application context information for critical mission executed in MANETs is given. Besides, assuming trust evaluation process is well known to all the participators that is a prerequisite to ensure a successful implementation in distributed systems, such as MANETs. Therefore, internal attackers may exploit all possible ways to sabotage the trust evaluation process. Robustness of trust system design must also be given great attention.

2.5 Conclusion

Trust-based routing concept is evolved from social science. This approach not only compensates the limitations of conventional secure routing schemes, but also simplifies the implementation requirements, which makes it more viable for reliable routing in MANETs. In general, trust-based routing scheme is a distributed prevention technique that is designed under the assumption that every node in the network knows how the system works. Therefore, when we design a trust-based routing scheme, we not only

need to handle those normal malicious actions as mentioned in Section 2.1, but also need to be aware of those compromised nodes that target the trust system itself. Each node inside the network will dynamically evaluate others' trust-level values based on its own experience and stores these trust information in its database. Each node may also disseminate these computed trust values to other interested peers to help others build up the trust information databases. Finally, each node can use these trust-level values in some decision making situations, such as routing path selection and attacker identification. In most trust-based routing schemes, they have to assume that each node has only one ID. The defense against an attacker with multiple faked IDs does not rely on the design of trust-based routing system, but on the authentication mechanism. Authentication is the first line of security defense against registering a faked ID. In other words, trust-based routing can be regarded as an add-on security component on top of the authentication mechanism. With a careful design of trust evaluation mechanism and authorized admission to the network, it is possible to achieve reliable routing and ensure network scalability under various adversary network environments. Finally, we show the completed unified framework of trust-based routing design in Figure 2.7. If a designed trust system only pays attention to the data packet or control packet loss, this system is only targeted on the abnormal packet dropping attack. If a designed system is able to verify the integrity of all important fields of the header of control message, it is capable of identifying modification attack. Nevertheless, in order to avoid the forged reply attack, the designed system not only needs to identify faked information contained in the header of control message, but also needs to authenticate node identity. At last, if a system can realize that the amount of RREQ messages in the network is above the normal level, we

claim that the system is able to prevent resource consumption executed by RREQ_AF attackers. The comparison of existing trust-based routing schemes is shown in Table 2.4.

Figure 2.7 Complete framework of trust routing system.

Table 2.4: Comparison of trust-based routing schemes.

Author	Trust Formulation			Trust Diss.	Decision Making	Attack Considered
	Source of Information	Information Type	Trust Metric			
S. Bucheger [32]	Direct Observation	N/D	Counting Policy	Not Addressed	Individual Node	Single Passive black hole
F. D. Rango [54]	Direct Observation & Request Feedback	B/C	Uncertainty	Requestor	Individual Node	RREQ_MF RREP_DR RREP_MF
T. Ghosh [35]	Direct Observation & Indirect Observation	B/C	Uncertainty	Direct Neighbors	Source Node	RREQ_AF RREP_MF RREQ_MF

Author	Trust Formulation			Trust Diss.	Decision Making	Attack Considered
	Source of Information	Information Type	Trust Metric			
W. Gong [36]	Direct Observation & Indirect Observation	B/C	Counting Policy	Direct Neighbors	Source Node	RREQ_DR RREP_DR Data packet dropping attack
H. Weerasinghe [39]	Policy based	N/D	Counting Policy	Not Addressed	Individual Node	RREP_FR RREP_MF RREQ_AF
B. Yang [34]	Direct Observation & Indirect Observation	B/C	Probability	Requestor	Individual Node	Single and Cooperative data packet dropping attack
A. Pirzada [31]	Direct Observation	B/C	Counting Policy	Next Hop (AODV)	Individual Node (AODV) Source Node (DSR)	RREQ_MF RREP_MF Data packet dropping attack
P. Michiardi [42]	Direct Observation & Indirect Observation	B/C	Probability	Direct Neighbors	Individual Node	RREQ_DR RREP_DR Data packet dropping attack
C. Zouridaki [43]	Direct Observation	B/C	Uncertainty	Next Hop	Source Node	Data packet dropping attack
P. B. Velloso [51]	Direct Observation & Indirect Observation	B/C	Counting Policy	Direct Neighbors	Individual Node	Single and cooperative data packet dropping attack
P. C. Tsou [40]	Policy-based	N/D	Counting Policy	Not Addressed	Individual Node	RREP_FR RREP_MF

P: Positive event; **N:** Negative event; **B:** Both positive and negative events

C: Continuous value; **D:** Discrete value

Chapter 3

Self-checking Trust Routing Scheme against Active Black Hole Attacks

In this Chapter, we propose a self-checking trust routing scheme to enhance reliable routing under adversarial MANET environment. In this proposed scheme we assume 1) all of the nodes in the network belong to the same authority and know how the system works. They all agree to cooperate and obey DSR routing regulation before admission, but some of them become misbehaved after entering the network. 2) Each node has only one ID. The defense of attacker with multiple faked IDs does not rely on the design of our self-checking trust routing protocol, but the authentication scheme. By skillfully collecting trust evaluation evidence, our proposed scheme can effectively prevent various forms of active black hole attackers. Even the internal attackers who know the security mechanism cannot get any benefit to launch those attacks. The rest of this chapter is organized as follows. Section 3.1 reviews existing defenses which utilize trust concept rather than cryptography to prevent active black hole attacks. Section 3.2 presents the proposed self-checking trust scheme in detail. Then we analyze the security aspects of proposed scheme against *internal* active black hole attackers in Section 3.3. Detection of colluded active black hole attackers is elaborated in Section 3.4. Section 3.5 presents simulation methodology and results. Finally, we conclude this chapter in Section 3.6.

3.1 Introduction

In Section 2.1 we have elaborated that attackers can achieve their attacking goals by misusing the DSR routing message. Among those typical security threats against normal routing, black hole attackers that aim to achieve route invasion is a popular one [55] and they bring more severe impact on network performance compared with other types of attackers, such as route disruption and resource consumption attacks. From Section 2.1 we know that black hole attacks can be divided into passive black hole attacks and active black hole attacks based on the ways of implementation. In *passive black hole attacks*, an attacker behaves well during route discovery process but discards all passing-by data packets once it is selected for data transmission. As the attacker only behaves uncooperatively but do not inject any false information into networks, this type of attack is known as passive routing attack. In *active black hole attacks*, an attacker always claims that it has the shortest path to the requested destination through modifying received route reply from others or injecting false route reply message directly. Once the attackers achieve route invasion, they can easily drop a lot of passing-by data packets without being known. From Figure 3.1 we can observe that packet delivery ratio decreases dramatically with the presence of a small portion of black hole attackers in the network. Since active black hole attackers attract and drop more data packets than the passive one, packet delivery ratio is lower with the presence of active black hole attacks. This motivates us to propose a self-checking trust scheme on top of original DSR routing protocols to prevent active black hole attacks.

Figure 3.1 Passive versus active black hole attack.

As we mention previously, in order to reduce computational overhead and sustain the improvised nature of MANETs, defense mechanisms which utilize trust concept rather than cryptographic technique becomes a new approach to prevent active black hole attacks. Usually, these defense mechanisms use pre-designed policies to determine the existence of attacker, and then try to identify the attacker. We can classify this type of defenses into following categories: 1) acknowledgement based schemes, 2) next hop indication schemes, 3) bait detection schemes, 4) monitoring based schemes, 5) multi-path based schemes.

3.1.1 Acknowledgement (ACK) Based Scheme

ACK-based schemes use required feedback to examine whether there is any attacker presence in the network. Receiving a reliable ACK message is the essential part for the execution of this type of defense mechanism. However, the open nature of MANETs allows all nodes to know the network configuration. Therefore, it can be assumed that compromised nodes also understand how the defense mechanism works. As a result, compromised nodes can successfully attack the system by either generating a faked ACK

or modifying the legal feedback received from others. Most of the time, ACK-based schemes [29, 56, 57] still require to use cryptography to ensure the integrity of the feedback message. These deficiencies are also summarized in Section 2.3.1.1, second paragraph. Besides, ACK based schemes are reactive that means nodes can only realize the existence of attackers after they are being attacked, but are not able to prevent in advance.

3.1.2 Next Hop Indication Scheme

Next hop indication schemes require intermediate node that generates a route reply (RREP) to indicate identity of next hop to the destination [38, 39]. In these schemes only those route replies generated by the destination are regarded as reliable. Route replies initiated by those intermediate nodes will not be accepted, unless they can pass the further checking. This type of defense mechanisms relies on a pre-agreed policy to implement intrusion detection component. As such, these defense schemes are also vulnerable to those internal attackers who break the pre-agreements. For example a compromised node can pretend they are only in charge of forwarding a route reply generated by the destination without indicating any required information in the reply message. Then the whole system may collapse. A typical example and its deficiencies have been described in Section 2.3.1.1, policy-based examination paragraph.

3.1.3 Bait Detection Scheme

In [40], P. C. Tsou et al. proposed a secure routing protocol named Baited-Black-hole DSR (BDSR) that aimed to identify black hole attackers. In BDSR, before source node initiates route discovery, it broadcasts bait (RREQ') that piggybacks a direct neighbor as requested destination. This bait RREQ' attracts attacker to send a faked route

reply (RREP). When a node receives the RREP, it gets a set of different addresses by comparing the replied path ($P = \{n_1, \dots, n_k, \dots, n_m, \dots, n_r\}$) and addresses in saved RREQ' ($Q'_k = \{n_1, \dots, n_k\}$). After that, both replied path (P) and the set of different addresses ($D_k = P \setminus Q'_k = \{n_{k+1}, \dots, n_m, \dots, n_r\}$) are reversed back to the source node. Once the source node gets D_k of every RREP forwarder, it can derive the segment location ($S = D_1 \cap D_2 \dots \cap D_k$) of attacker and the temporary trust set ($T = P \setminus S$). Finally, the source node will ask the second last node in temporary trust set T to monitor to which address the last node in T sends the packet to. According to reported monitoring result, source node can identify the attacker. For example, in Figure 3.2 source node n_1 pretends to request a route to direct neighbor n_6 by broadcasting a RREQ'. When n_3 receives the faked route reply from n_4 , it derives the set of different addresses D_3 , and forwards both D_3 and RREP back to the source node. Likewise, when n_2 receives the RREP, it performs the same operation. When n_1 receives the RREP, D_2 and D_3 , it can obtain the temporary trust set T . After that source node will request n_2 to monitor to which node n_3 may send the packets. According to the monitoring result of n_2 , source node can identify n_4 is an attacker.

However, if attacker n_4 knows the prevention scheme, it can pretend it only helps forwarding a route reply generated by n_5 , and also appends a set of different addresses ($D_4 = \{n_5, n_6\}$). Thus, attacker misleads source node in getting a false temporary trust set ($T = \{n_1, n_2, n_3, n_4\}$). After that source node will ask n_3 to monitor packet forwarding state of n_4 . Even if n_4 forwards the packet along the faked path P , the packet is still lost, since next hop n_5 is a faked address added by attacker n_4 . However, source node may consider the faked node n_5 as an attacker.

Figure 3.2 Example of bait detection scheme.

3.1.4 Monitoring Based Scheme

This type of schemes rely on overhearing to monitor the forwarding states of neighboring nodes and evaluate forwarders' trust value according to the monitoring results [31]. At the same time, monitoring node may disseminate these trust values to help peers to obtain more trust information about the network [28, 36]. After that nodes can use these trust values to distinguish malicious and benevolent parties.

However, these monitoring based schemes suffer from several problems. Firstly, nodes have to keep receiving packets from their neighbors in order to monitor packet forwarding status, thus it will cost a lot of energy. Secondly, each node can only monitor forwarding status of neighboring nodes, which does not ensure the forwarded packets can be received by the third node. For example, if active black hole attackers injected some faked routing information during route discovery, even if they helped forwarding the data packets instead of dropping, the packets were still lost. Thirdly, in many existing trust based schemes, trust information is required to exchange between nodes to help each other to infer the accurate trust values. However, if ratings made by others are considered, the trust system may be vulnerable to false accusation or false praise executed by internal

attackers [14]. Fourthly, most monitoring based trust schemes are not collusion resistant. Besides, overhearing is not always available for all wireless interfaces, such as in an asymmetric-link environment. Lastly, similar to ACK based schemes, these trust based schemes are also reactive, which are not an ideal solution to prevent active black hole attackers.

3.1.5 Multipath Based Scheme

Multipath based schemes use redundant routing paths to avoid attackers [58-61]. In [58], the authors proposed to find more than one path from a source to its destination and only those paths with some shared nodes are considered as reliable paths for data transmissions. In [59], the authors introduced a Learning Automata Based Fault-Tolerant Routing Algorithm (LAFTRA) that uses learning automata module to optimize the selection of path by avoiding faulty nodes. To find the optimal routing path, every node relies on the ACK message from the destination to evaluate the goodness value of current route. Finally, the node could choose the most optimal path for data transmission based on the goodness values of routing paths.

The major issue of these multipath based schemes is the lack of malicious node detection. These schemes can avoid routes containing attacker and those that are broken, but without making any distinction between these two cases and without further information regarding the node causing the packet loss. Furthermore, many multipath-based schemes [60, 61] need to work together with ACK based and trust based schemes, thus they consequently inherit those drawbacks mentioned above.

3.2 Self-checking Trust Routing Scheme

We propose a self-checking trust scheme (SCTS) on top of DSR protocol to overcome those deficiencies of the existing prevention schemes as mentioned in Section 3.1. There are three basic operations of SCTS: *updating/maintaining neighborhood-topology*, *liar checking* and *consistency checking*. In this scheme, all nodes work independently based on their own assessment. Furthermore, each node maintains the following records:

- *Neighborhood-Topology Table (NTT)*: contains the most updated neighborhood connectivity information.
- *Liar List*: stores the number of times a node has been caught due to supplying false neighbor information.
- *Inconsistency List*: stores the number of times a node sent a RREP, but the replied route was inconsistent with receiver's neighborhood-topology table.
- *Destination Cache*: when a node receives or generates a route request (RREQ), it stores the destination address into this cache before processing the request message.
- *Black List*: stores the node that has been identified as an attacker and the remaining time that the node need to stay in this black list. During data transmission, if node **X** finds next node **Y** in the black list, it launches *route maintenance* by sending a RERR to avoid packet loss. At the same time, node **X** will not help forwarding any packet from node **Y** to punish it.

3.2.1 Updating/Maintaining Neighborhood Topology

In our scheme every node broadcasts Hello message in its local area with adaptive frequency. The Hello message includes sender address (*Sender ID*) and information of its direct neighbors (*Neighbor ID[i]*), as shown in Table 3.1. By exchanging Hello messages, each node can obtain the neighborhood topology within two hops. When node **X** receives a Hello message that contains more than α direct neighbors, node **X** ignores this Hello message. Otherwise, node **X** updates one-hop neighbor and time to live (TTL) fields of its neighborhood-topology table (NTT), as shown in Table 3.2, according to *sender ID* containing in received Hello message. After that **X** launches the *liar checking* operation and updates two-hop neighbor fields of its NTT accordingly. The intuition behind α is that if the number of direct neighbors containing in the Hello message exceeds the maximum theoretical value; most probably the sender added some faked neighbor IDs into broadcast Hello message. We will explain the procedure of liar checking later in Section 3.2.2. Figure 3.3 shows the procedure of updating/maintaining NTT.

Table 3.1: Hello message format.

Sender ID
Neighbor ID[1]
Neighbor ID[2]
...
Neighbor ID[n]

Table 3.2 Neighborhood topology table.

one-hop neighbor[1]			TTL[1]
two-hop neighbor[1.1]	two-hop neighbor[1.2]	...	
one-hop neighbor[2]			TTL[2]
two-hop neighbor[2.1]	two-hop neighbor[2.2]	...	
...			
one-hop neighbor[n]			TTL[n]
two-hop neighbor[n.1]	two-hop neighbor[n.2]	...

Figure 3.3 Updating/Maintaining neighborhood-topology.

For example, when node **X** receives node **Y**'s Hello message for the first time, which contains three neighbors **A**, **B** and **X**. Node **X** puts node **Y** into one of the empty one-hop neighbor fields of its NTT and sets TTL field to an initial value. If both reported neighbor **A** and **B** pass the liar checking operation executed by node **X**, reported neighbor **A** and **B** become node **X**'s two hop neighbors that are put into corresponding

two-hop neighbor fields. The value in TTL field will be automatically deducted by one per second, if node **X** does not receive any further Hello message from node **Y**. Once the value in this field becomes 0, the corresponding one-hop and two-hop neighbors that contain in node **X**'s NTT will be removed. If node **Y** has already existed as a one-hop neighbor in the NTT of node **X**, node **X** only updates TTL field and two-hop neighbor fields that can be reached through node **Y**.

Each node maintains their latest two hops topology in their NTTs. If certain node ID can be found in both one-hop neighbor[i] and two-hop neighbor [j,k], this implies one-hop neighbor[i], one-hop neighbor[j] and the owner of this NTT are connected together. Every node will use the latest neighboring topology that stored in their NTTs to execute consistency checking upon receiving a route reply (RREP) message.

3.2.2 Liar Checking

When a node receives a Hello message, before updating those reported neighbors into corresponding two-hop neighbor fields of its NTT, it will execute liar checking operation. For example, node **Y** informs node **X** that node **A** is a direct neighbor of itself. Node **X** will check whether it can find node **A** in its *destination cache*. If node **A** exists, which means node **X** has received/generated certain RREQ that asking a route to node **A** before. Therefore, node **X** will pay more attention and further check whether it can find node **A** in any one-hop neighbor field of its NTT. If it finds and node **A** did not report node **Y** as a symmetric neighbor, node **X** puts **Y** into *liar list* and increases node **Y**'s lying-count by one. If the lying-count of node **Y** reaches the threshold β , node **Y** will be put into black list by node **X**. The complete liar checking procedure is shown in Figure 3.4.

Figure 3.4 Liar checking procedure.

3.2.3 Consistency Checking

Once a node collected the latest neighborhood topology via *updating/maintaining neighborhood-topology* and filtered out suspicious information through *liar checking*, it fulfills the prerequisite to operate *consistency checking*.

The initial route discovery process in our SCTS is slightly different from that in DSR. When source node needs a route to destination, it initiates a route discovery by broadcasting a route request message (RREQ), and stores requested destination address into *destination cache*. Any node receives this RREQ will also keep the requested destination address into *destination cache*. After that, they either keep broadcasting this

RREQ, if they do not have a route to the requested destination, or send a route reply (RREP) back to the source. Normally, every intermediate node helps to forward any received route reply back to the source along the reverse path that can be fetched from the accumulated record of route request message, and source node accepts all received route replies. Finally, the source node will choose the shortest route as final routing path for data transmission. In an unsecured environment, our primary goal is to ensure every routing path is free of black hole nodes. Each node is responsible for judging whether the previous node who sends a route reply (RREP) is an attacker or not. Then the receiver decides whether to keep processing this received route reply based on its judgment. The procedure of consistency checking is illustrated in Figure 3.5.

For instance, node **X** receives a RREP from node **Y** that contains a path **S-X-Y-D**. Firstly, node **X** will try to find node **Y** in any one-hop neighbor[i] field of its NTT, then it randomly picks up ρ nodes from two-hop neighbor[i, j] fields and check whether it can find node **D** in these ρ nodes. If node **X** finds node **D** in these randomly selected ρ nodes, which means the received reply path is consistent with node **X**'s NTT, node **X** keeps forwarding this RREP back to the source node. Otherwise, node **X** will drop this RREP, put node **Y** into inconsistency list and increase inconsistency-count of node **Y**. Once the inconsistency-count of node **Y** reaches a threshold γ , node **X** will put **Y** into its blacklist for a certain amount of time denoted as T_{BL} . The intuition of ρ is that if the number of node **Y**'s direct neighbor is more than the average number, node **X** only regards ρ nodes as its two-hop neighbors that can be reached via node **Y**. Through this way, even if node **Y** randomly claims additional neighbors in previously broadcasted Hello message, most probably node **X** can detect the inconsistency when node **Y** sends a faked RREP. In order to avoid checking error accumulation and ensure the flexibility of our scheme, we

maintain liar list and inconsistency list separately. Since the possibility to catch the lying behavior of a node is smaller than that of detecting an inconsistency reply generated by a suspicious node, we define separate thresholds β and γ for lying-count and inconsistency-count, respectively.

Figure 3.5 Consistency checking.

Self-checking refers to the fact that each node executes consistency checking to ensure received RREP is congruous with its neighborhood topology. If a node declares that it is directly connected with the requested destination through a RREP message and it did not claim the requested destination was a one-hop neighbor before, this node is most probably an attacker. In summary, this self-checking scheme is distributed. Each node independently executes liar checking and consistency checking with the help of its

latest neighborhood topology. If certain node fails either liar checking or consistency checking several times, it will be identified as an attacker and put into the blacklist.

3.3 Security against internal attacks

This section analyzes the security aspects of our proposed scheme in terms of defending against internal attackers. If internal active black hole attackers know how the self-checking scheme works, they can add some faked neighbors into broadcast Hello messages. Two types of lying behaviors may exist as follows:

- *Internal Attacker Type A*: besides the real neighbors, the attacker always randomly adds some faked direct neighbors into periodically broadcast Hello message. When the attacker receives a RREQ, it behaves as an active black hole attacker that always replies “I am directly connected with the destination.”
- *Internal Attacker Type B*: when an attacker receives a RREQ that asking a route to certain destination (e.g. node **D**), it immediately broadcasts a Hello message that include this requested destination (e.g. node **D**) as a direct neighbor, and always includes **D** as the direct neighbor in later broadcast Hello messages. Once the attacker receives a RREQ that asking a route for the same destination again, it replies that it is directly connected with node **D**.

In the following parts, we will explain how SCTS can resist these internal attackers.

3.3.1 Internal Attacker Type A

In our self-checking scheme, the defense mechanism against internal attacker type A has two perspectives. Firstly, when a node receives a Hello message that contains more than α number of direct neighbors, this Hello message will be dropped directly. Secondly,

before verifying the consistency between received RREP and NTT, if receiver finds the sender/forwarder of a RREP has more than ρ direct neighbors ($\rho < \alpha$), receiver only randomly chooses ρ nodes as its two-hop neighbors. Since the attacker does not know who will be the requested destination in the future, and it can only randomly add maximum α nodes into broadcast Hello message, the chance that the attacker just includes the future destination is rare. In other words, internal attacker type A is very likely to be caught during consistency checking.

In [62], C. Bettstetter et al. pointed out that the node distribution has a peak in the center of design region under random waypoint model. From [62] we can know that the probability density function, $f_{X,Y}(x, y)$, of a node's location (X, Y) on a $1000 \times 1000m^2$ square region is independent of moving speed and it can be approximated as:

$$f_{X,Y}(x, y) = \frac{36}{x_m^3 y_m^3} xy(x - x_m)(y - y_m) \quad 0 < x, y \leq x_m \quad (3.1)$$

x : Coordination in the X-axis; y : Coordination in the Y-axis

x_m : Maximum value in X-axis; y_m : Maximum value in Y-axis

In this square region, $x_m = y_m = 1000m$. We can get the cumulative distribution function of (X, Y) by taking an integration of equation (3.1), which is shown in below.

$$\begin{aligned} F_{X,Y}(x, y) &= \int_0^x \int_0^y f_{X,Y}(x, y) dx dy \\ &= \frac{1}{x_m^3 y_m^3} x^2 y^2 (4xy - 6xy_m - 6yx_m + 9x_m y_m) \quad (3.2) \end{aligned}$$

Probability of finding a node in a region $(0, x_1) \times (0, y_1)$ is $Pr(0 < X \leq x_1, 0 < Y \leq y_1) = F_{X,Y}(x_1, y_1)$. Since in our example transmission coverage of a node is $\pi \times 250^2 m^2$, approximate to a $443 \times 443 m^2$ square. The probability to find a node in this $443 \times 443 m^2$ square that located in the center of design region as shown in Figure 3.6 is given by equation (3.3)

$$\begin{aligned}
 & \Pr(278.5m < X \leq 721.5m, 278.5m < Y \leq 721.5m) \quad (3.3) \\
 &= F_{X,Y}(721.5, 721.5) - F_{X,Y}(721.5, 278.5) - F_{X,Y}(278.5, 721.5) + F_{X,Y}(278.5, 278.5) \\
 &\approx 0.388094
 \end{aligned}$$

Therefore, if there are 50 nodes with the same transmission range 250m and each of them moves according to the random waypoint model in a $1000 \times 1000m^2$ design region, the number of nodes in a shadow region as shown in Figure 3.6 is $\lceil 50 \times 0.388094 \rceil = 20$. In other words, a node has 20 neighbors at most, when it positioned in the center of the design region. As a result, we can set α to 20.

Figure 3.6 Region with maximum number of nodes.

On the other hand, if all nodes are uniformly distributed in a design region, once the node knows the area of the network and the total number of nodes, it can roughly estimate the average number of neighbors around itself based on equation (3.4):

$$\rho = \left\lceil N \div \frac{\text{Area of design region}}{\text{Transmission coverage}} \right\rceil \quad (3.4)$$

where N is the total number of participated nodes and ρ is the average number of direct neighbors. For example, there are total of 50 nodes with the same maximum transmission range of 250m. They are uniformly distributed on a $1000 \times 1000m^2$ design region and move according to the random way point model. Based on equation (3.4) we get:

$$\rho = \left\lceil 50 \div \frac{1000 \times 1000}{\pi \times 250^2} \right\rceil = 10 \quad (3.5)$$

This means, on average each node has 10 neighbors around itself. As shown in Figure 3.7, we take snapshots at different time of simulation and show the average number of direct neighbors around all nodes. We can observe that the average number is fluctuated around 10, which is consistent with our calculation. If the value of ρ is set too small, a lot of RREP that are generated/forwarded by benevolent nodes may be dropped during consistency checking. If the value of ρ is set too large, it increases the probability for the internal attacker type A to pass the consistency checking.

Figure 3.7 Average number of neighbors.

3.3.2 Internal Attacker Type B

The defense mechanism against internal attacker type B relies on the *liar checking* process. We use Figure 3.8 as an example to illustrate how this defense mechanism works. For example, node **S** broadcasts a RREQ asking a route to destination **D**. When attacker **A** receives this RREQ, it immediately broadcasts a Hello message to include **D** as its direct neighbor. Although node **S** cannot detect any abnormality and will update its

NTT according to this received Hello message, node **C** and **E** can realize attacker **A** is lying. Since node **C** and **E** are directly connected with node **D**, through checking their NTTs, node **C** and **E** can realize that node **D** did not declare node **A** as a symmetric neighbor in the past, they will ignore this reported neighbor **D** by node **A** and increase the lying-count of **A** by one separately. Due to the mobility of networks, majority of nodes will eventually be able to identify the liar.

Figure 3.8 A simple network topology.

3.4 Enhanced Self-checking Trust Routing Scheme

We have a solution to identify single and internal active black hole attackers by comparing received RREP with the latest information stored in NTT. However, if two attackers act in coordination, they can still escape from being caught. For example, attacker **A₁** and **A₂** are colluded together as shown in Figure 3.9. When **A₁** receives a route request (RREQ) that asking a route to destination **D**, instead of declaring it is directly connected with node **D**, **A₁** sends a faked reply like **S-A₁-A₂-D** back to the **S**. Since **A₁** colluded with attacker **A₂**, even **A₂** realizes **A₁** faked a connection against itself it keeps silent. In such a situation node **S** cannot realize any abnormality by verifying this route reply with information stored in its NTT, since node **S** only knows two hops topology around itself.

Figure 3.9 Colluded active black hole.

In order to prevent such a collusion attack, we augment periodically exchanged Hello message to include not only direct neighbors but also two-hop neighbors as shown in Table 3.3. Correspondingly, the NTT of each node contains not only two-hop neighbors, but also three-hop neighbors. The detail operations of *updating/maintaining neighborhood topology*, *liar checking* are the same as we mentioned before. During the *consistency checking*, any node receives a route reply will verify the consistency among not only two-hop neighbors in the route reply message and its NTT, but also three-hop neighbors. For example, as shown in Figure 3.9, node S receives a replied path S-A₁-A₂-D from node A₁. First node S tries to find A₁ from any one-hop neighbor[i] field of its NTT. After that node S randomly selects ρ nodes from two-hop neighbor[i, j] and removes the rest. At last, with the assist of NTT, node S verifies whether A₁ has reported node D can be reached through A₂ before. If the replied path is not consistent with information stored in NTT, node S will drop this RREP, put A₁ into inconsistency list and increase inconsistency-count of A₁. Similarly, the attacker A₁ can also extend the faked RREP like S-A₁-A₂-A₃-D to avoid being caught. In other words, the distance between attacker A₁ and destination D is now three hops away in the faked RREP. For the sake of simplicity, let us denote the scenario where the attacker is n-hops away from the destination as AD_n, where n is an integer.

Table 3.3 Extended Hello message format.

Option Type		Sender ID	
one-hop neighbor[1]			
two-hop neighbor[1.1]	two-hop neighbor[1.2]	...	two-hop neighbor[1.n]
one-hop neighbor[2]			
two-hop neighbor[2.1]	two-hop neighbor[2.2]	...	two-hop neighbor[2.n]
⋮			
one-hop neighbor[n]			
two-hop neighbor[n.1]	two-hop neighbor[n.2]	...	two-hop neighbor[n.n]

If the faked RREP route length is too long, the attacker in fact cannot gain any ascendancy to achieve route invasion anymore. To prove this point, we use the same simulation settings shown in Table 3.4. When an attacker receives a RREQ and finds the requested destination is not reachable within three hops, it immediately sends a faked reply to claim that it can reach the destination in three hops. Through this way we can count the amount of the packet loss in AD3. Packet loss in AD2 and AD1 can also be collected in a similar procedure. As can be seen from Figure 3.10, almost all packet loss is caused due to those faked paths in AD1. Only 18% of packet loss is caused by those faked reply paths in AD2. However, less than 3% of packet loss is caused by those faked paths in AD3. This is because the faked reply paths may even longer than other real routes. So the packet loss in AD3 is mainly due to the attackers that are really in the routing path and drop data packets passing by. Thus, we focus on AD2 in this section. In addition, we only study the effect that internal attackers add some faked direct neighbors into broadcast Hello message. From the attacker's point of view, adding some faked direct neighbors is more attractive. But if necessary, same countermeasure can be applied to against attackers add faked two-hop neighbors into Hello messages.

Figure 3.10 Percentage of packet loss due to faked path.

3.5 Simulation Model and Results

To demonstrate the performance of our proposed scheme, we use NS-2 [63] which is an event driven simulator to simulate our network environment. IEEE 802.11 is used as the underlying medium access control (MAC) layer protocol. The default simulation parameters are listed in Table 3.4.

Table 3.4 Simulation parameters.

Simulation area	1000m x 1000m	Interface queue	20
Number of nodes	50	Default simulation time	900
Default number of attacker	10	Frequency of exchanged 'Hello'	Every 2s
Default Max Speed (V_{max})	10m/s	Time to Live (TTL) in NTT	4s
Pause time	0s	Max number of direct neighbor(α)	20
Transmission range	250m	Average number of direct neighbor(ρ)	10
Default Traffic flow pairs	10	Inconsistency-count threshold(γ)	10
Traffic type	CBR (UDP)	Lying-count threshold(β)	5
Packet length	512 bytes/packet	Release from blacklist T_{BL}	100s
Packet generation rate	4 packets/s	The number of faked neighbors that are added by internal attacker type A	10

We use a rectangular space of size $1000m \times 1000m$ with 50 nodes uniformly distributed inside this region. The number of attackers is set to 10. When simulation starts, every

node moves according to a random waypoint model with a randomly chosen speed uniformly distributed between 0m/s and V_{max} . Default V_{max} is set to 10m/s. All nodes keep moving by setting *pause time* to 0s. The transmission coverage of each node is simulated as a circular area with maximum transmission range of 250m. There are 10 traffic flow pairs. Source node uses constant bit rate (CBR) that sends 512-byte data packet every 0.25s. Each node has an interface queue for packets awaiting transmission that holds up to 20 packets. Once the simulation is on, every node starts exchanging Hello message every two seconds, the initial value of TTL field in NTT is set to 4s. By default, the simulation lasts 900s. Inconsistency-count threshold γ and lying-count threshold β are set to 10 and 5, respectively. A node will be identified as an attacker and put into black list if either the accumulated inconsistency-count or lying-count reaches the threshold. Node is released from black list after T_{BL} , which is set to 100s. According to Section 3.3.1, in this simulated environment, we set ρ (average number of direct neighbors) to 10 and α (maximum number of direct neighbors) to 20. All simulations are performed 20 rounds for each configuration and the average results are presented with a confidence interval of 95%.

3.5.1 Attacker Pattern

In our simulation, various active black hole attacks have been considered:

- *Single Active Black Hole (SABH)*: Once it receives a RREQ message, it immediately replies “I am directly connected to the destination.”
- *Colluded Active Black Hole (CABH)*: When attacker receives a route request (RREQ) and it finds the requested destination is not reachable within one hop, at the same time there is another attacker exists in its transmission range, it will launch a

collusion attack by sending a reply to deceive others that it has a routing path through itself, collusion partner, and then reach the destination. Otherwise, it just behaves as a single active black hole.

- *Internal Attacker Type A*: Besides the real neighbors, attacker randomly adds some faked direct neighbors into periodically broadcast Hello messages. When the attacker receives a route request, it behaves as a single active black hole attacker. The default number of randomly added direct neighbors is set to 10.
- *Internal Attacker Type B*: When attacker receives a RREQ that asking a route to certain destination (e.g. node **D**), it immediately broadcasts a Hello message that includes this requested destination (e.g. node **D**) as a direct neighbor, and keep including node **D** in later broadcast Hello messages. Once the attacker receives a RREQ that asking a route to the destination **D** again, it behaves as a single active black hole.
- *Mixed Internal Attacker (MIA)*: Besides the real neighbors, attacker not only claims the overheard destination as a direct neighbor but also randomly adds some faked direct neighbor into broadcast Hello message. The total number of false neighbors in a Hello message is set to 10.
- *Mixed Active Black Hole (MABH)*: if attacker finds a collusion partner in its transmission range, it launches collusion attack. Otherwise, the attacker randomly decides whether it behaves as a single active black hole, internal attacker type A, or internal attacker type B.

3.5.2 Performance Metrics

To evaluate the performance of the protocols, we use the following metrics:

- *Packet delivery ratio (PDR)*: It is the ratio of the number of data packets received by the destination nodes to the number of packets sent by the source nodes.
- *Throughput*: This gives the average number of bits received by the destinations per second.
- *Normalized Routing overhead*: This is the ratio of the total number of control packets generated (including Hello message) to the total number of data packets received.
- *End-to-end delay*: This defines the average time that a data packet takes to traverse the distance between the source and the destination.

3.5.3 Baseline and Benchmark

In our simulations, the basic DSR described in Section 1.3 without any prevention mechanism is used as a baseline. It is expected that the baseline system may perform badly in most of the adversarial situations.

Trust vector model DSR (TVDSR) [36] is chosen as a benchmark. Same as our self-checking scheme, this protocol is also designed under assumption that all nodes know how the scheme works. As its name indicates, TVDSR belongs to the monitoring based scheme that we mentioned in Section 3.1.4. In TVDSR, trust vector of node **X** to node **Y** consists of three elements which are evaluation of experience, knowledge and recommendations. This vector is shown as equation (3.6):

$$V(X \rightarrow Y) = [E_{X,Y} \ K_{X,Y} \ R_{X,Y}] \quad (3.6)$$

Experience evaluation $E_{X,Y}$ is node **X**'s evaluation to node **Y** by monitoring packet forwarding state of node **Y**. This evaluation measures the packet forwarding ability of node **Y**. Knowledge evaluation $K_{X,Y}$ is node **X**'s evaluation to node **Y** by direct observing MAC layer link quality. It measures the probability that data packet will be transmitted

successfully between them. Recommendation evaluation $R_{X,Y}$ is node $\mathbf{X}$'s evaluation to node $\mathbf{Y}$ by collecting recommendation about node $\mathbf{Y}$ from neighboring nodes. The final trust value node $\mathbf{X}$ assigns to node $\mathbf{Y}$ is defined as:

$$T_{X,Y} = W_E \times E_{X,Y} + W_K \times K_{X,Y} + W_R \times R_{X,Y} \quad (3.7)$$

$$(\{W_E, W_K, W_R\} \in [0,1], \{E_{X,Y}, K_{X,Y}, R_{X,Y}\} \in [0,1])$$

where $[W_E, W_K, W_R]$ is a trust policy vector composed of three weight elements according to the three-dimensional trust vector V . Finally, this normalized trust value will be integrated with DSR to find a routing path with the highest trust value instead of the shortest path.

3.5.4 Simulation Results

We evaluate the performance of our self-checking trust scheme (SCTS) and enhance version (ESCTS) with the following tests:

Test 1-Effect of Number of Attackers

In this test we compare our proposed SCTS and ESCTS with DSR under the same unsecure environment. From Figure 3.11, we can observe that by varying the number of attackers we can observe PDR drops sharply with the presence of only 5 active black hole attackers (DSR-SABH), followed by the colluded active black hole attackers (DSR-CABH). This is because the faked reply paths that are generated by the single active black hole attackers are shorter than the colluded nodes' replies. Single active black hole attackers can attract more data packets passing by and drop them silently. This results the lower packet delivery ratio. As the number of attackers increases, the situation becomes worse. Fortunately, even with the present of 15 mixed attackers, ESCTS-MABH can still achieve a PDR of 60%-70%. Since our scheme can ensure more data packets being

received by the destination under adversarial environment, the throughput also increases as shown in Figure 3.12.

Figure 3.11 Test 1-Number of attacker versus PDR.

Figure 3.12 Test 1-Number of attackers versus throughput.

However, our ESCTS achieves a high PDR and throughput at the expense of increased routing overhead and end-to-end delay as shown in Figure 3.13 and Figure 3.14, respectively. In our scheme, we require periodically broadcast Hello message to

obtain the latest neighboring information, this is one of the main reason that causes the higher routing overhead compared to the baseline. For ESCTS-CABH scheme, since a receiver does not know whether collusion attack is launched or not, any node related to any mismatch between two hops away or three hops away node in the replied route and receiver's NTT will be regarded as an attacker. This will cause some detection errors and more route error (RERR) messages will be disseminated. Therefore, the ESCTS always shows a higher routing overhead compared with the SCTS. For DSR-CABH, since attackers need to explore the available collusion partner before launch the attack, periodically broadcast Hello messages are also required. Thus, the routing overhead of DSR-CABH is also higher than the DSR-SABH. When nodes execute ESCTS, they try to avoid attacker and user more secure path instead of the shortest one. Therefore, we can observe that the end-to-end delay in our ESCTS increases with more attackers inside the network.

Figure 3.13 Test 1-Number of attackers versus routing overhead.

Figure 3.14 Test 1-Number of attackers versus delay.

Test 2-Effect of Mobility

Figure 3.15 shows the effect of mobility on the performance of our proposed schemes. It can be seen that both SCTS and ESCTS perform better under low mobility. This is because high mobility will affect the accuracy of the NTT. This table can only provide the neighboring topology in the past. In a high mobility environment, the neighboring topology may change frequently. This will cause detection errors when the node utilizes the information stored in its NTT to verify received route reply messages. In addition, we can notice ESCTS-CABH is more sensitive to the mobility change than the SCTS-SABH. This is again because a receiver in ESCTS-CABH does not know whether collusion attack is launched or not, thus any node related to any mismatch between two hops away or three hops away node in a replied route and receiver's NTT will be regarded as an attacker.

Figure 3.15 Test 2-Maximum speed versus PDR.

Test 3-Robustness of ESCTS

In this section we present the study on our ESCTS against internal attacker type A, internal attacker type B and mixed internal attacker, separately. In Figure 3.16, if we remove all the countermeasure mechanisms against internal attacker type A as we mentioned in Section 3.3.1, we can observe that the PDR decreases linearly with the increase of the number of faked direct neighbors that contained in each Hello message. If attackers only randomly add five faked neighbors into each broadcast Hello message, the packet delivery ratio only degrades less than 10% compared to the scenario that no faked neighbor is added. If internal attackers randomly add more faked neighbors (e.g. twenty faked neighbors), their chance to get a correct guess of the future destination and intrude into data routing path will increase, the packet delivery ratio degrades nearly 40%. Our proposed scheme can quickly filter out those Hello messages containing too many neighbors (e.g. more than 20 direct neighbors). As can be seen from Figure 3.16, the performances of ESCTS on those internal attackers with 10, 15 or 20 additional faked

neighbors are even better than that without internal attackers. This is because the more faked neighbors internal attackers added into broadcast Hello messages, the easier they will be caught. If attackers only randomly add few faked neighbors, they have small chance to guess the future destination correctly. In other words, attackers have little chance to achieve route invasion due to our *consistency checking* operation. At the same time, our random trust (e.g. RREP receiver randomly chooses 10 nodes as its two-hop neighbors) operation will also help to some extent.

Figure 3.16 Test 3-ESCTS against internal attacker type A.

Figure 3.17 shows the performance of ESCTS against internal attacker type B. At the beginning, this type of internal attackers can intrude into data routing paths and degrade the performance. As time goes by, more and more nodes can realize the lying behavior of attackers through liar-checking operation. PDR increases as the simulation time elapses. Finally, we want to show the effect of mixed attack which combines two types of internal attacks as shown in Fig. 3.18. Although mixed internal attackers add both previously overhead destination nodes and some faked direct neighbors into

periodically broadcast Hello message, the defense principle is exactly the same as we described to prevent two types of internal attackers. It can be seen that under mixed internal attackers, the performance of our ESCTS is comparable with the presence of internal attacker type A or type B.

Figure 3.17 Test 3-ESCTS against internal attacker type B.

Figure 3.18 Test 3-ESCTS versus varying internal attacker.

Test 4-Sensitivity of ESCTS

In ESCTS, inconsistency-count is used to denote the number of times a node sent a route reply that is inconsistent with receiver's neighboring topology table. This kind of inconsistency is usually caused by the SABH. Lying-count is used to record how many times internal attacker type B has been caught lying. Therefore, the setting of inconsistency count threshold and lying-count threshold may affect the performance of our scheme against SABH and internal attacker type B separately. If the threshold values are too small, it may cause false detection that identifies a benevolent node as an attacker. On the contrary, if the threshold value is too large, it will take a longer time to catch an attacker. Table 3.5 shows the influence of different thresholds on the PDR of our scheme against these two types of attacks. From Table 3.5, we can observe our proposed scheme is not sensitive to the setting of thresholds.

Table 3.5 Test4-PDR versus varying threshold values.

Inconsistency-count Threshold (γ)	5	10	15
ESCS-SABH	0.6722	0.7555	0.7444
Lying-count Threshold (β)	3	5	7
ESCS-Internal Attacker Type B	0.6221	0.6566	0.6534

Test 5-Performance of ESCTS under lossy environment

To model a lossy environment, we assign a random number uniformly distributed between a pre-defined value X and 1 for each link. In this study, X is chosen to be [0.7, 0.8, 0.9, 1.0] as shown in Figure 3.19. If X is set to 1, it refers to an idea network environment without any link loss. Fig. 3.19 shows that the improvement between our proposed scheme and original DSR decreases as the link quality becomes worse. Since lossy environment affects the accuracy of obtained neighboring topology, it affects the accuracy of attacker detection process. Under an adversarial environment, as long as the

link quality higher than 0.9, our ESCTS scheme can maintain a considerable PDR improvement.

Figure 3.19 Test 5-Performance evaluation under lossy environment.

Test 6-Comparison with TVDSR

As the TVDSR [36] is used to detect malicious packet drop by SABH attack only, we restrict our test to SABH and use the exactly same parameter settings in TVDSR. Table 3.6 indicates those parameters that are different from our default settings. Since we change the simulation area to $670 \times 670\text{m}$, we need to reset the average number of direct neighbor (ρ) and maximum number of direct number (α) to 22 and 36, respectively. These values can be computed based on our explanations in Section 3.3.1.

As can be seen from Figure 3.20, both TVDSR and ESCTS can successfully detect the active black hole attacks. However, unlike TVDSR, our ESCTS can catch those attackers before data transmission rather than identify them after they drop data packets. Thus, when the number of attackers inside the network is less than half, our ESCTS outperforms TVDSR 10%-20% in terms of PDR. When the number of attackers

increases, most of the time the attackers may be really in the routes but just drop data packets passing by. In this situation, our ESCTS has no advantage comparing to TVDSR. In addition, due to the periodically broadcast Hello messages our ESCTS brings more routing overhead than TVDSR as shown in Figure 3.21.

Table 3.6 Specific simulation parameters in test 6.

Simulation area	670m x 670m
Simulation time	600s
Max speed (V_{max})	20m/s
Pause time	10s
Max number of direct neighbor(α)	36
Average number of direct neighbor(ρ)	22

Figure 3.20 Test 6-Packet delivery ratio comparison.

Figure 3.21 Test 6-Routing overhead comparison.

3.6 Conclusion

A proactive security routing protocol and its enhanced version were proposed with the necessary assumption that internal attackers have the knowledge about how the prevention mechanism works in MANETs. Our scheme could be applied on top of the conventional routing protocols as a complimentary security measure. The key idea is that every node is required to exchange neighbor information before route discovery, and then uses previously collected neighboring information to verify each received route reply. As the attackers do not know who will be the requested destination in next route request message, they have no idea what neighbor information they need to fake in order to avoid being caught. If they randomly add many faked neighbors into broadcast Hello messages, they can be easily identified. If certain node refuses to exchange neighbor information, definitely it will be caught if it behaves as an active black hole attacker in next second. If attackers provide faked neighbor information after they know the

requested destination, *liar-checking* operation will function. By utilizing previously collected neighboring information we can greatly increase the robustness of our prevention system. Besides identifying internal and colluded attackers, our proposed self-checking trust scheme also has the following salient features: 1) In our scheme all nodes only believe first-hand information that is obtained and verified by themselves. They execute trust evaluation for other nodes independently. Through this way, we can avoid lots of internal attacks that target on the trust system itself, such as slandering, self-promoting attacks [14]. 2) Since only negative evidences are collected in our proposed trust evaluation, active grey-hole attackers cannot get any advantage by behaving well and badly alternatively. As long as they launch active black hole attack, their trust records will decrease. Once the accumulated trust record breaks the threshold, the grey-hole attacker will be isolated. Although grey-hole attackers may not be isolated by the system immediately due to their on-off behavior, it is rare for those faked reply messages to pass the consistency checking that is designed in proposed scheme. In other word, grey-hole attacker cannot bring any impact to our proposed scheme and will be identified sooner or later. 3) Different from the existing monitoring and acknowledgement based schemes that can only identify active black attackers after they drop packets, our self-checking trust scheme attempt to identify attackers before they attack the system. 4) This self-checking scheme does not rely on any extra hardware or computational intensive cryptographic tools. It can be easily applied on existing routing protocols with minor modifications, which makes it suitable for practical implementations in MANETs. Extensive simulations confirmed the effectiveness of the proposed self-checking trust scheme, where its enhanced version could further improves the security at the expense of increased routing overhead and end-to-end delay.

Chapter 4

An Evolutionary Self-Cooperative Trust Scheme against Passive Attacks

In Section 3 we proposed a self-checking trust scheme to prevent active black hole attacks in MANETs. Comparing to the active black hole attackers, passive black hole attackers' behaviors are more covert to be identified. Since the attackers are really in the routing path and only discard passing-by data packets silently, all the proactive defense mechanisms and cryptographic techniques are not able to help under passive attacks network environment. In Figure 3.1 we shown that with the presence of 10% passive active attackers, they cause around 50% degradation of network performance. All these reasons motivate us to develop this evolutionary self-cooperative trust scheme (ETS) on top of the normal ad hoc routing protocols to enhance reliable routing under passive attacks network environment. The rest of this chapter is organized as follows. Section 4.1 presents a brief review of related works and provides an overview of our ETS mechanism. Section 4.2 explains the proposed ETS in detail. Simulation model and results are presented in Section 4.3. Finally, we conclude this chapter in Section 4.4.

4.1 Introduction

Secure routing plays an important role in reliable network performance. However, most of the conventional secure routing protocols relied on cryptographic tools [64-66] are not feasible in MANETs as we have mentioned in Section 2.2.3. The most important issue in these secure routing protocols is that they cannot identify those passive attacks, such as *passive black hole attacks* and *grey-hole attacks*. Consequently, trust based routing schemes become a new approach to ensure reliable routing in MANETs. In general, trust based routing schemes that try to mitigate and stimulate misbehaved or uncooperative nodes can be classified into two groups [67]: reputation based approach and credit-based approach.

Reputation Based Approach

In most reputation based schemes, each node relies on overhearing to monitor the packet forwarding status of neighboring nodes and evaluates forwarders' reputation levels according to the monitoring results [31]. Since some researchers also consider this technique as a reactive method to prevent active black hole attacks, we have already introduced the implementation and deficiencies of this prevention mechanism in Section 3.1.4.

Credit-Based Approach

The second type of cooperative enforcement mechanisms are based on credit system or game theory. In these schemes [68, 69], they treat the transmission of a packet from source to destination as a deal. Nodes offering service will be rewarded and nodes receiving service will be charged. The credit to be rewarded or charged is determined by a credit service center, in which the nodes track a receipt for the received message.

During cooperation, nodes report their actions of receiving/forwarding by uploading the receipt to the credit service center, after that credits will be issued accordingly. Besides the credit service center, a tamper-proof mechanism is also required to ensure the integrity of uploaded receipts. However, these requirements may reduce the applicability of credit system in an open and improvised network, such as MANETs. In [70], game theory is used to incentive grey-hole nodes to help messages propagation. It is similar to the credit system that nodes may get different payoffs with different actions (e.g., whether they help others to forward packet, when they offer help to others etc.). Different players have their own sets of strategies in game theory, and they always try to maximize the profits by adjusting their own strategies. Finally, an equilibrium state in the network can be derived, where no players can change their strategies to gain more payoffs. Both credit system and game theory approach are designed under assumption that all nodes, including attackers, are rational players. Their objectives are to maximize their own profits. However, if attackers behave as terrorists without caring about any payoff or rewards, these defense techniques may not be effective anymore. Besides, these credit-based approaches focus on stimulating the cooperation rather than identifying the attackers.

In order to overcome those deficiencies of the existing trust based routing schemes [36, 46, 70, 71] as mentioned above, we propose an evolutionary self-cooperative trust scheme (ETS) that can be easily implemented on top of the existing ad hoc routing protocols. As the name indicates, this scheme is based on two pillars. On the one hand, every node executes self-detection to evaluate the trust levels of other nodes (benign or malicious) independently. On the other hand, they share self-detection results with their direct neighbors to help other nodes to get more trust information about the network.

Each node puts more confidence in those trust information (properties of peers) that are explored through self-detection. Furthermore, self-detection experience will influence the attitude of node (optimistic or pessimistic) towards the network, and consequently affect its decision when it receives shared trust information from neighbors. Compared to the existing trust based routing schemes, the proposed ETS has the following features:

- 1) Researchers intuitively take it for granted that high mobility makes the reliable routing become more difficult to achieve in MANETs. In this chapter, we demonstrate that mobility can in fact be helpful to enhance security.
- 2) We develop a generic trust evaluation system that can be easily applied on existing routing protocols. This ETS is capable of preventing not only conventional passive black hole attacks but also those internal attackers who even know how the security mechanism works. To the best of our knowledge, few secure routing schemes have taken internal attackers into consideration.
- 3) The ETS mechanism requires direct neighbors to exchange trust information periodically. But the final trust level node N_x assigns to node N_y (benign or malicious) is affected by not only the number of votes N_x receives from its neighbors, but also its attitude (optimistic or pessimistic) towards the network environment.
- 4) Each node puts more confidence in those trust information explored by self-detection than the one derived through shared results from neighbors. And these trust information can only be modified when the node obtains new information with higher or equal privilege level. Each node may explore diverse forms of trust information, and correct their cognition about the network evolutionarily.

4.2 Evolutionary Self-Cooperative Trust Scheme (ETS)

This ETS is implemented on top of the DSR protocol to enhance reliable routing with the presence of various passive attackers. In this scheme we make the same assumption that we have mentioned in Chapter 3, which are as follows. 1) All of the nodes in the network belong to the same authority and know how the system works. They all agree to cooperate and obey DSR routing regulation before admission, but some of them become misbehaved after entering the network. 2) Each node has only one ID. The defense of attacker with multiple faked IDs does not rely on the design of our self-checking trust routing protocol, but the authentication scheme. Authentication is the first line of security that makes registering a new or faked ID difficult.

ETS consists of two parts, which are *self-detection* and *cooperative-detection*. In our scheme, each node runs *self-detection* independently, and then broadcasts detection results that indicate benign and malicious peers to its direct neighbors. After that, based on *self-detection* and information received from neighbors, each node can perform *cooperative detection* to derive more trust information to distinguish malicious and benign nodes. In particular, *cooperative detection* only serves as a complement to *self-detection*, and it can never override *self-detection*. To facilitate the operation of ETS scheme, every node broadcasts Hello messages in its coverage with adaptive frequency. This Hello message includes the sender's ID and its most recently discovered results via self-detection. With the help of Hello messages, each node can obtain the IDs of its direct neighbors timely. This information is helpful in *self-detection*. At the same time, shared trust information containing in the Hello messages will help the execution of *cooperative*

detection. Besides the periodically broadcast Hello messages, each node also maintains the following types of records:

- 1) *Sending History Cache (SHC)*: the number of data packets it generated or forwarded along a specified routing path.
- 2) *Receiving History Cache (RHC)*: the number of data packets it received as a destination along a specified routing path.
- 3) *Self-Analysis Record (SAR)*: the cooperative and uncooperative counts of a particular node.
- 4) *Neighbor Judgment Record (NJR)*: the direct neighbor IDs and the latest trust information that is shared by these direct neighbors.
- 5) *Trust Information Record (TIR)*: the trust levels of other nodes and the privilege level of each piece of trust level information. There are two possible trust levels, 1 and -1, representing benign and malicious node, respectively. In addition, trust information obtained through self-detection and cooperative detection will be assigned to different privilege levels like primary and secondary, represented by 1 and 0, respectively. In brief, trust level indicates the properties of peers (benign or malicious), and the privilege level reveals whether this piece of trust level information is obtained via self-detection or cooperative detection. TIR can only be overwritten when a new piece of trust information comes with higher or equal privilege level. Initially, the trust levels of other nodes are unknown and their privilege levels are set to 0. This design is consistent with our common sense that we always give more confident for information obtained via ourselves rather than heard from others. Even the information obtained via ourselves is incorrect we may still believe it first and gradually correct it with the improvement of self-cognition.

The first three records are used in self-detection and the fourth one is used in cooperative detection. Each node uses TIR to store the self-detection and cooperative detection results dynamically. During data transmissions, if node N_x finds next node N_y is indicated as malicious node in the TIR, it launches *route maintenance*. At the same time, node N_x will not help forwarding any packet from node N_y so as to punish it. The design details of self-detection and cooperative detection will be discussed in the rest of this chapter.

4.2.1 Self-Detection

The idea of self-detection essentially resembles human behavior for which the most secure way to obtain information from a person is by approaching him/her directly. Since every node is mobile in MANETs, it is possible for every two nodes to become direct neighbors. Once a source or intermediate node N_i realizes a past destination N_D becomes a direct neighbor, it sends an *investigation request* (IREQ) immediately to check how many data packet node N_D received along a specified source route (e.g., SR1) in the past. Based on the *investigation reply* (IREP) from N_D , node N_i can judge whether there are any hidden malicious nodes between itself and N_D in SR1. If node N_i transmitted m packets using source route SR1 and all the packets had been received by node N_D , which implies all intermediate nodes between N_i and N_D in SR1 were benign. Node N_i updates its SAR by adding m to the cooperative counts of all intermediate nodes along SR1, respectively. However, if N_i finds m packets were lost, and there were t intermediate nodes between itself and node N_D in SR1, node N_i will increase $\lceil m/t \rceil$ equally to the uncooperative counts of all those intermediate nodes along SR1. Because node N_i cannot identify which intermediate node caused the packet loss, it can only punish all suspicious

peers equally to avoid omission of any malicious peer. After node N_i collects these cooperative and uncooperative counts, it will utilize Bayesian probability to combine these evidences to a usable trust metric [72]:

$$\begin{cases} P_{i,j}^b = \frac{C_{i,j}+1}{C_{i,j}+U_{i,j}+2} \\ P_{i,j}^m = \frac{U_{i,j}+1}{C_{i,j}+U_{i,j}+2} \end{cases} \quad (4.1)$$

where $P_{i,j}^b$ is the probability that node N_i believes N_j will behave cooperatively in next packet forwarding process, and $P_{i,j}^m$ is the probability that node N_i thinks node N_j will not cooperate in the next packet forwarding process. $C_{i,j}$ and $U_{i,j}$ are the cooperative and uncooperative counts that N_i assigned to N_j . If $P_{i,j}^b$ or $P_{i,j}^m$ is greater than a self-detection threshold α , node N_i will update its *TIR* by indicating N_j as a benign or malicious node accordingly, and set the privilege level of this piece of trust information as primary.

As aforementioned, periodically broadcast Hello messages can help nodes to know which past destination has become a direct neighbor at the current moment. Besides this, when a source or intermediate node generates or forwards a data message, as long as it is not directly connected with the destination it needs to update its *SHC*. When a destination node receives a data packet, it will update its *RHC*. For example, the *SHC* and *RHC* of node N_x may look like Table 4.1 and Table 4.2. *Total packet* field in *SHC* and *RHC* records the amount of data packets N_x has forwarded/received along the specified source route. Time stamp records the most recent time when the corresponding entries are updated. When node N_c encounters N_x and sends an *IREQ* to ask how many data packets N_x received through a specified route (e.g., B-A-C-F-X) in the past, node N_x will search its *RHC*, as show in Table 4.2 and reply an *IREP* back to node N_c . At the

same time, node N_x copies this replied value that is extracted from total packet field into last investigation reply field of its RHC accordingly. When N_c compares this received IREP and the record that stored in its SHC, it can calculate the trust metric of the intermediate node between itself and the destination (e.g., node N_F) based on equation 4.1. After that, node N_c will remove this entry from its SHC to conserve memory space. If the SHC or RHC of a node is full, the oldest entry will be removed. The procedure of self-detection is shown in Figure 4.1.

Table 4.1 SHC of node N_x .

Destination	Source Route	Total Packet	Time Stamp(s)
B	F-X-C-E-B	32	12.4s
B	F-X-A-B	359	68.4s
D	X-A-B-P-D	120	68.4s

Table 4.2 RHC of node N_x .

Source Route	Total Packet	Time Stamp(s)	Last Investigation Reply				
B-A-C-F-X	76	24.5s	B	0	A	0	C 76
F-P-A-X	102	84.6s	F	0	P	0	
S-L-K-X	355	84.6s	S	0	L	0

Figure 4.1 Self-detection procedure.

4.2.2 Cooperative Detection

To perform cooperative detection, each node has to collect trust information from its direct neighbors. Therefore, when node N_x receives a Hello message containing the sender's most recent self-detection results, it will update its NJR as shown in Table 4.3. However, if node N_x finds a piece of trust information from neighbor N_y against the self-detection result stored in its TIR, node N_x will discard this piece of information. This is because neighboring shared trust information only services as a complement, but cannot override self-detection result. The value in time to live (TTL) field of Table 4.3 will be

deducted by one per second, if node N_x does not receive any further Hello message from that specified one-hop neighbor. Once the value in this field becomes zero, corresponding benign and malicious nodes that are reported by this neighbor will be removed. When node N_x collects enough trust information from its neighbors, together with its self-detection results extracted from its TIR, it can generate a utility matrix for cooperative detection. In the rest of this section, we will use an example to explain how to use the utility matrix and how much trust information is considered enough to perform the cooperative detection.

Table 4.3 Neighbor judgment record (NJR).

One-hop neighbor [1]				TTL [1]
Benign node [1.1]	Benign node [1.2]	...	Benign node [1.n]	
Malicious node [1.1]	Malicious node [1.2]	...	Malicious node [1.n]	
One-hop neighbor [2]				TTL [2]
Benign node [2.1]	Benign node [2.2]	...	Benign node [2.n]	
Malicious node [2.1]	Malicious node [2.2]	...	Malicious node [2.n]	
⋮				
One-hop neighbor [n]				TTL [n]
Benign node [n.1]	Benign node [n.2]	...	Benign node [n.n]	
Malicious node [n.1]	Malicious node [n.2]	...	Malicious node [n.n]	

Table 4.4 Utility matrix of node N_x .

Examiner		N ₂	N ₃	N ₄	N ₈	N ₉	1: Benign -1: Malicious
	N _x	1	?	?	?	-1	
	N ₁		1	1			
	N ₂				1		
	N ₃	1		1	-1		
	N ₄	1	1			-1	
	Examinee						

For example, the utility matrix of node N_x looks like Table 4.4. The first row is the self-detection results that are extracted from TIR of N_x . The following rows show the collected trust information from neighbors, which can be fetched from NJR. From the matrix shown in Table 4.4, we can see that examiner N_x graded N_2 and N_9 as benign and malicious peers, respectively. Examiner N_1 graded both N_3 and N_4 as benign peers. The rest rows in this matrix read similar. Most of the cells in this matrix are empty since each examiner can only identify few examinees' trust levels (benign or malicious) through self-detection. The problem is that node N_x wants to utilize these sparse data in the utility matrix to infer the missing entries indicated by question marks. Inspired by the collaborative filtering technique that is applied in some recommendation systems, like Netflix [73], we exploit all the data in the utility matrix, trying to discover structures in the pattern and values of the entries in this matrix. Before giving the detail, we list some notations in Table 4.5.

Table 4.5 Summary of notations for cooperative detection.

Notations	Descriptions
a_i	Attitude of examiner N_i
b_j	Behavior of examinee N_j
$\hat{G}_{i,j}$	Inferred grade an examiner N_i assigns to examinee N_j , which is obtained through cooperative detection.
$G_{i,j}$	Actual grade examiner N_i assigned to examinee N_j , which is shown in the utility matrix
A	Constant matrix
c	A given vector composed of all $G_{i,j}$ in the utility matrix
z	Variable vector composed by all a_i and b_i

Firstly, we introduce two parameters: a_i to model the attitude of each examiner, and b_j to model the behavior of each examinee. Therefore, the inferred grade an examiner $\mathbf{N}_i$ assigns to an examinee $\mathbf{N}_j$ is expressed:

$$\hat{G}_{i,j} = a_i + b_j \quad (4.2)$$

In fact, the idea of this model emulates human psychology. For example, if most of the peers that $\mathbf{N}_i$ identified through self-detection in the past are malicious, the attitude of node $\mathbf{N}_i$ towards the network may become pessimistic, the value of a_i may become very negative, and vice versa. In addition, if many neighbors of $\mathbf{N}_i$ report that node $\mathbf{N}_j$ is benign, the value of b_j will approach to a positive value. However, whether $\mathbf{N}_i$ will grade $\mathbf{N}_j$ as a benign node is affected by two parameters a_i and b_j . Now the problem can be simplified to solving the least square problem. When node $\mathbf{N}_x$ solves the parameter a_x and all parameters b_j based on the utility matrix, it can infer those grades indicated by question marks. The least square problem in our application is equivalent to:

$$\text{minimize}(\hat{G}_{i,j} - G_{i,j})^2 \quad (4.3)$$

$$\text{subject to: number of } a_i \text{ and } b_j < \text{number of } G_{i,j}$$

where $G_{i,j}$ is the actual grade examiner $\mathbf{N}_i$ assigned to $\mathbf{N}_j$, which is shown in the utility matrix. $\hat{G}_{i,j}$ is the inferred grade based on equation (4.2). Therefore, the minimization is over all the $M+N$ parameters, where M is the number of examiners and N is the number of examinees in the utility matrix. We can rewrite this minimization function (4.3) in the standard form of a least square problem:

$$\text{minimize} \|Az - c\|^2 \quad (4.4)$$

$$\text{subject to: elements in vector } z < \text{elements in vector } c$$

where A is a known constant matrix, c is a given vector composed of all $G_{i,j}$ and variable vector z contains all the $M+N$ parameters in the utility matrix. For example, the problem in Table 4.4 is to find all parameters that can achieve global minimization of the following quadratic function:

$$\begin{aligned} & (a_x + b_2 - G_{x,2})^2 + (a_x + b_9 - G_{x,9})^2 + (a_1 + b_3 - G_{1,3})^2 + (a_1 + b_4 - G_{1,4})^2 + \\ & (a_2 + b_8 - G_{2,8})^2 + (a_3 + b_2 - G_{3,2})^2 + (a_3 + b_4 - G_{3,4})^2 + (a_3 + b_8 - G_{3,8})^2 + \\ & (a_4 + b_2 - G_{4,2})^2 + (a_4 + b_3 - G_{4,3})^2 + (a_4 + b_9 - G_{4,9})^2 \end{aligned} \quad (4.5)$$

This function (4.5) can be easily transferred to a standard least square problem. The given vector $c = [G_{x,2}, G_{x,9}, G_{1,3}, G_{1,4}, G_{2,8}, G_{3,2}, G_{3,4}, G_{3,8}, G_{4,2}, G_{4,3}, G_{4,9}]^T$, variable vector $z = [a_x, a_1, a_2, a_3, a_4, b_2, b_3, b_4, b_8, b_9]^T$, and the matrix A is an 11×10 constant matrix with each entry being 0 unless the corresponding examiner graded the corresponding examinee, in which case it is 1. For example, the first row of matrix A that corresponds to N_x graded N_2 as shown in Table 4.4 is expressed $[1, 0, 0, 0, 0, 1, 0, 0, 0, 0]$.

In order to find the global minimization of function (4.4), we can take the first derivatives of function (4.4) with respect to variable z and set them to zero.

$$2A^T A z - 2A^T c = 0 \quad (4.6)$$

We know this is a minimizer because the objective function is convex. After that, we can solve the variable vector z

$$z = [A^T A]^{-1} * A^T c \quad (4.7)$$

In order to avoid multiple solutions to the above linear equation (4.7), we have to ensure the number of all parameters is less than the number of all known grades in the utility matrix. *This is the very reason that it requires each node to collect sufficient trust information before performing cooperative detection.* Once N_x solves the variable vector z , it can infer the trust levels of those unknown nodes based on equation (4.2). If

$\hat{G}_{x,j}$ is greater than 0.5, N_x will indicate N_j as a benign node represented by 1. If $\hat{G}_{x,j}$ is smaller than -0.5, N_x will indicate N_j as a malicious node represented by -1. Otherwise, the property of the node is still unknown. The procedure of cooperative detection is shown in Figure 4.2.

Figure 4.2 Cooperative detection procedure.

As we mentioned before, cooperative detection is a complement to the self-detection, all trust levels of nodes that are inferred through cooperative detection will be

assigned to secondary privilege level. Once the node collects some new trust information from its neighbors or through self-detection, it will perform a cooperative detection procedure again and update its TIR. If a node does not explore any trust information through self-detection before, the cooperative detection result will purely result in the parameter b_j .

4.3 Simulation and Results

We use NS-2 to evaluate the performance of ETS scheme. DSR is used as the underlying routing protocol based on IEEE 802.11 MAC layer protocol. The simulation parameters are listed in Table 4.6.

Table 4.6 Simulation parameters.

Simulation area	1000m x 1000m
Number of nodes	50
Default No. of attacker	20
Default Max Speed (V_{max})	20m/s
Pause time	0s
Transmission range	250m
Default traffic flow pairs	10
Traffic type	CBR(UDP)
Packet length	512bytes
Packet arrival rate	4 packets/s
Interface queue size	20
Default simulation time	2000s
Frequency of exchanged 'Hello'	Every 1s
Time to Live (TTL) in NJR	4s
Self-detection threshold (α)	0.7
Size of SHC and RHC (Total number of source routes can be contained)	600

This simulated MANET occupies a $1000\text{m} \times 1000\text{m}$ region with 50 nodes uniformly distributed inside this region. The number of attackers is set to 20. Random walk model is employed for node mobility. Specifically, when the simulation starts, every node moves toward a randomly selected direction with a randomly chosen speed uniformly distributed between 0m/s and $V_{\max}$. In our simulation, $V_{\max}$ is set to 20m/s and all nodes keep moving by setting pause time to 0s . The transmission coverage of each mobile node is simulated as a circular area with maximum transmission range of 250m . There are 10 traffic flow pairs. A source node adopts constant bit rate (CBR) that sends 512-byte data packet every 0.25 second. Each node has an interface queue holding up to 20 packets awaiting transmission, and all packets are managed in a drop tail model. Once the simulation is on, every node starts exchanging Hello messages every one second. The initial value of time to live (TTL) field in NJR is set to 4 seconds. By default, each simulation run will last 2000 seconds. Self-detection threshold α is set to 0.7. Since the trust metric represented in Section 4.2.1 equation (4.1) is similar to the definition of packet forwarding ratio that measures the ratio between the number of forward data packets and the total number of expected forward data packets. For a normal peer, the packet forwarding ratio should be greater than 0.5 and less than 1, thus we choose 0.7 as the self-detection threshold. Each node can contain up to 600 different source route records in its SHC and RHC, respectively. If the SHC or RHC is full, when a new message comes in, the oldest entry will be removed automatically. All results are presented with a confidence interval of 95%.

4.3.1 Attack Patterns

In our simulation, various passive routing attacks have been studied as follows:

- *Passive Black Hole (PBH)*: A malicious node participates devotedly in the route discovery process, but dumps all data packets which it is supposed to help forwarding.
- *Grey-hole (GH)*: The behavior of grey-hole node is similar to passive black hole. However, instead of dropping all data packets, it selectively forwards data packets. In our simulation, we assume the probability that a grey-hole node may help other nodes to forward a data packet is 30%.
- *Mixed Routing Disruption (MRD)*: A malicious node randomly decides whether it behaves as an active black hole, passive black hole or grey-hole node.
- *Malicious Collective (MC)*: This type of attack refers to the internal attack, such that all the attackers know how the ETS scheme works. We assume all attackers are colluded together before entering the network. When an attacker encounters a new peer that is not belonging to this collective, it can easily identify that peer as benign. Attacker from this collective will always broadcast other malicious peers as benign and those discovered benign nodes as malicious.

4.3.2 Performance Metrics

To evaluate the performance of the protocols, we use the following metrics:

- *Packet Delivery Ratio (PDR)*: It is the ratio of the number of data packets received by the destination nodes to the number of packets sent by the source nodes.
- *Normalized Routing Overhead*: This is the ratio of the total number of generated control packets (including Hello message, IREQ and IREP) to the total number of received data packets.

- *End-to-end Delay*: This defines the average time a data packet takes to traverse the distance between the source and the destination.
- *Throughput*: This gives the average number of bits received by the destinations per second.

4.3.3 Baseline and Benchmark

In this simulation, we also use DSR that is described in Section 1.3.1 as a baseline. It is expected that the baseline system may perform badly in most of the adversarial situations.

Trusted DSR (TDSR) [31] is chosen as a benchmark. Same as our scheme, TDSR also targets on identifying the passive attacks and designed under assumption that all nodes know how the scheme works. In TDSR, every node executes trust analysis by the trust agent independently. For example, when a node N_x transmits a packet, such as route request (Rq), route reply (Rp), route error (Re) and data message (Dm), it buffers that packet for a specified time interval and monitors the behavior of neighboring node N_y via overhearing. If node N_y forwards the packet properly within the specified time interval, the successful count C_s of that type of packet (Rq/Rp/Re/Dm) is increased. Otherwise, the failure count C_f will be increased. After node N_x collects these counting numbers, which are the trust evidences, it can compute the trust level of N_y based on the following equations:

$$C(i) = \begin{cases} \frac{C_s - C_f}{C_s + C_f}, & \text{if } C_s + C_f \neq 0 \\ 0, & \text{otherwise} \end{cases} \quad (4.8)$$

$$T_{xy} = \sum_{i=1}^m [W(i) \times C(i)] \quad (4.9)$$

where $C(i)$ represents one type of the packet being considered for trust computation. Different types of packets will be assigned different weights, $W(i)$, based on the requirement of the application. The final trust value that node N_x assigns to N_y is represented by T_{xy} that is derived from all types of packets. In TDSR, each node only computes trust value of other nodes that had direct experience before. Since nodes are mobile, the neighboring nodes also keep changing. After some time the node may get most of the trust information of other peers inside the network. Finally, the source node will choose the route with the highest trust value instead of the shortest path.

4.3.4 Simulation Results and Analysis

We evaluate our proposed ETS under various routing scenarios: 1) Test 1 to Test 3: to evaluate the performance of ETS against normal passive routing attackers under various situations. 2) Test 4 to Test 6: to evaluate the robustness of ETS scheme against malicious collective. 3) Test 7: to study the performance of ETS under lossy network environment. 4) Test 8: to compare ETS scheme with TDSR.

Test 1-Effect of Number of Attackers

In this set of test, we show the results of the comparison of the proposed ETS and DSR under the same unsecure environment. From Figure 4.3 and Figure 4.4, we can observe that PDR and throughput drop drastically with the presence of only 5 passive black hole attackers (DSR-PBH), followed by the grey-hole (DSR-GH). As the number of attackers increases, the situation becomes worse. Fortunately, even with the presence of 40% attackers inside the network, our ETS can still achieve a PDR of 70%. In particular, for the passive black hole attacks, our ETS increases the PDR more than 150% as compared with basic DSR. Since our scheme can ensure more data packets being

received by the destination nodes under the adversarial environment, the throughput also increases as shown in Figure 4.4.

Figure 4.3 Test 1-Number of attacker versus PDR.

Figure 4.4 Test 1-Number of attacker versus throughput.

However, our ETS scheme achieves the high PDR and throughput at the expense of increased normalized routing overhead and end-to-end delay. In our scheme, we require periodically broadcast Hello messages to discover current neighbors and share self-detection results. In addition, we introduce the investigation request/reply control packets for self-detection. These additional control packets are the main reason that causes the increased normalized routing overhead as shown in Figure 4.5.

When nodes execute ETS, they try to avoid those attackers and use a secure path instead of the shortest one. Therefore, we can observe that the end-to-end delay of our ETS increases when there are more attackers inside the network as shown in Figure 4.6. However, the end-to-end delay of DSR decreases when there are more attackers. This is because in the presence of more attackers, most data packets cannot be received by the destinations but being dropped by the attackers. All these lost packets have not been included into the end-to-end delay measurement.

Figure 4.5 Test 1-Number of attacker versus routing overhead.

Figure 4.6 Test 1-Number of attacker versus delay.

Test 2-Effect of mobility

Figure 4.7 shows the effect of mobility on the performance of ETS scheme. It can be seen that the PDR of our proposed scheme improves with the increasing mobility. This improvement can be attributed to the improved probability of attacker detection. High mobility environment causes more node interactions, more nodes may happen to meet as direct neighbors and perform self-detection. When maximum speed reaches 30m/s, no matter which type of routing disruption attacker is present, our ETS scheme can increase the PDR more than 90% compared to the basic DSR. Contrary to our intuition, mobility is in fact helpful to enhance network security in ETS, and this further proves the applicability of ETS in MANETs.

Nothing is free. High mobility environment can improve the detection rate and result in high PDR with the presence of attackers, but more memories of sending history cache are required to store previous routing records. In Figure 4.8, we present the

average overflow time of the sending history caches with increasing mobility. From this figure, we can convince that under high mobility environment, more source routes will be recorded due to frequently topology change, which results a faster overflow of sending history caches. Under our simulation settings, since receiving history caches do not overflow even under high mobility environment, we do not present the result.

Figure 4.7 Test 2-Maximum speed versus PDR.

Figure 4.8 Test 2-Maximum speed versus cache overflow.

Test 3-Effect of Size of Simulation area

Size of simulation area also affects the performance of the ETS and baseline scheme under an adversarial networking environment. In Figure 4.9 we can observe that ETS can increase the PDR regardless of simulation area, but the amount of improvement compared to the baseline is the most remarkable when the side length of simulation area is between 800m and 1000m. This is because when the network area becomes smaller, a source node is more likely to find the requested destination as its direct neighbor, and routing attackers have small chance to intrude into the routing path. As we can see in Figure 4.9, when the side length of simulation area reduces to 400m, even under unsecured environment, basic DSR performs fairly well. However, when the network area becomes larger, the performance of the ETS scheme degrades. This is because on the one hand in a large simulation area it becomes difficult for a node to encounter a past destination node. On the other hand, the routing path becomes longer in a large network area, thus it may cause some self-detection errors. As we mentioned in Section 4.2.1, in the self-detection each node would rather scarify some benign peers than omit any malicious peer. When the routing path become longer, there are more intermediate nodes may be involved between the self-detection executor and destination. If the executor realizes some packet loss from destination's reply, it will increase the uncooperative count of all intermediate nodes along this path equally. This may scarify some benign peers and cause some self-detection errors.

In Figure 4.10, we also present the average number of hops between source node and destination node with increasing simulation areas. From this figure it can be clearly observe that when the length of simulation area reduces to 400m, source nodes are almost directly connected with destination. Therefore, even under unsecured network

environment, the performance degradation is not obvious. When the side length of simulation area is between 800m and 1000m, there are about 3 to 4 hops between source and destination in average. In other words, when an intermediate node encounters the destination and executes the self-detection, it can easily distinguish those benign and malicious nodes with minimum errors. This ensures the most remarkable PDR improvement when the side length of simulation area is between 800m and 1000m as shown in Figure 4.9. When the side length of simulation area becomes larger, the routing path becomes longer as well.

Figure 4.9 Test 3-Simulation area versus PDR.

Figure 4.10 Test 3-Simulation area versus hops.

Test 4-Impact of malicious collective with simulation time

This test studies ETS against internal attackers. Those routing attackers, such as passive black hole and grey hole, not only disrupt normal routing but also attack our trust system by broadcasting faked trust information. In other words, each type of routing disruption attacker is combined with malicious collective. From Figure 4.11, we can observe that even with the presence of malicious collective ETS performs well and its PDR increases gradually as time elapses. Although self-promoting and slandering executed by malicious collective may cause some detection errors and degrade the performance of our ETS at the early stage, as time goes by, when every node explores trust levels of peers via self-detection, they can correct those detection errors and filter later received false trust information. Therefore, we claim that ETS scheme enables every node to correct their cognition about the network evolutionarily.

(a)

Figure 4.11 Test 4-Simulation time versus PDR under (a) GH and (b) PBH.

Test 5-Robustness of ETS

In order to test the robustness of ETS, we gradually increase the number of attackers inside the network and let attackers form the malicious collective. From Figure 4.12, we can observe that forming a malicious collective does not increase the trust levels of malicious nodes enough for them to bring much impact on our trust system. This is because voting concept implied in our cooperative detection ensures that as long as the majority nodes are honest, ETS can resist false trust information from malicious collective. Furthermore, as time goes by, self-detection will be able to correct and filter more and more false trust information. Self-detection provides a baseline to guarantee the accuracy of trust evaluation. Even in the presence of 30 attackers inside the network, our ETS can still improve the PDR by 50% as compared to basic DSR.

(a)

(b)

Figure 4.12 Test 5-Robustness test of ETS on (a) GH and (b) PBH.

Test 6-Accuracy rate of cooperative detection

Since a malicious collective may affect the performance of cooperative detection, we present the average accuracy rate of cooperative detection performed on those benign peers in the following tests. Figure 4.13 shows that without the interference from malicious collective, the average accuracy rate of cooperative detection maintains around 90% throughout the whole simulation. However, due to lack of trust information at the early time, benign nodes are misled by the faked information from malicious collective, this causes the low accuracy rate at initial state. Luckily, with the accumulation of self-detection experience and more trust information collected from benign peers, benign nodes become more intelligent to correct and filter those faked trust information, the average accuracy rate of cooperative detection increases as time elapses.

Figure 4.13 Test 6-Accuracy rate of cooperative detection under (a) GH and (b) PBH.

Test 7-Evaluation of ETS under lossy environment

We use the same model that has been introduced in Section 3.5.4 to evaluate the performance of our ETS with lossy links. In this test, we assign a random number uniformly distributed between a pre-defined value X and 1 for each link. The values of X are selected to be 0.7, 0.8, 0.9 and 1.0 as shown in Figure 4.14. If X is set to 1, it refers to an idea network environment without any link loss. Since lossy environment affects the accuracy of self-detection, it may further increase the detection error of cooperative detection. Therefore, we can observe that the improvement of our proposed scheme over original DSR decreases as the link quality becomes worse. On average there are only three hops between source node and destination node when the length of simulation area is 1000m as presented in Figure 4.10, as long as the link quality is higher than 0.8, our ETS can still maintain at least 40%-60% PDR improvement with the presence of any type of passive attackers.

Figure 4.14 Test 6-Link quality versus PDR.

Test 7- Comparison with TDSR

As the trusted DSR (TDSR) [31] is simulated under mixed routing disruption (MRD) attack environment, we also restrict our test to mixed routing attacks and use exactly the same parameter settings in TDSR. Except the simulation time, all other settings are the same as we present in Table 4.6. In this experiment we set the simulation time to 900s instead of 2000s to prove that ETS can quickly improve the performance even in a shorter time. From Figure 4.15, we can see that both TDSR and ETS can increase the PDR under mixed routing disruption attack environment. Different from TDSR that only uses direct observation to evaluate peers' trust levels; in our scheme shared trust information from neighboring nodes also contributes to the trust level analysis. Through exchanging trust information, ETS can speed up information propagation and mitigate trust information shortage issue. If we remove the cooperative detection mechanism mentioned in Section 4.2.2, self-detection mechanism can only slightly improve the PDR

compare to the basic DSR but performs worse than the TDSR as shown in Figure 4.15. Fortunately, once self-detection and cooperative detection mechanisms are integrated, ETS outperforms TDSR about 20%-35% in terms of PDR. Therefore, we can conclude that cooperative detection utilizing trust information from neighboring nodes contributes a lot to the trust level analysis in ETS scheme.

Figure 4.15 Test 7-Comparison with TDSR.

4.4 Conclusion

Mobile ad hoc networks (MANETs) have wide applications in practice and may bring a great revolution in our lives in the near future. However, we have to carefully resolve the security issues before their successful deployment. In this chapter, we propose an evolutionary self-cooperative trust scheme (ETS) to defend against various passive routing attackers. This ETS mechanism emulated human cognition can promote network scalability and ensure reliable routing in MANETs via three aspects. Firstly, self-detection mechanism utilizes mobility to perform trust analysis, which provides a

baseline that ensures the accuracy of trust evaluation. Secondly, cooperative detection mechanism can further improve the performance and enhance the robustness of this trust system. On the one hand, cooperative detection speeds up the trust evaluation process and mitigates trust information shortage issue. On the other hand, voting concept implied in the cooperative detection ensures that as long as the majority of the nodes inside the network are honest, forming a malicious collective will not increase the trust value of malicious peers enough in order for them to have impact on the network. Thirdly, evolutionary operation of ETS enables nodes to explore diverse form of networks, to alleviate the impact of false information provided by the dishonest entities and correct their cognition by themselves. As long as the majority of nodes inside the network are honest, this proposed ETS can improve the PDR more than 90% as compared to basic DSR regardless the types of routing attacks.

Noteworthy, although we only tested this ETS on top of DSR routing protocol, distributed nature of ETS mechanism allows it to be easily implemented on other routing protocols to handle various packet dropping attacks. Besides, if we require all nodes to periodically broadcast two-hop neighbors and integrate *consistency checking*, *liar checking* and *updating/maintaining neighborhood* mechanisms, which we have introduced in Chapter3, into ETS scheme, we can develop a defense system that is able to proactively prevent active black hole attacks and reactively identify those passive packet dropping attacks.

Chapter 5

Simulation Study of Attacks Under

Network Coding Environment

In Chapter 2 we study the security vulnerabilities of the normal ad hoc routing protocol, such as DSR. Then we proposed some trust-based routing mechanisms to handle these security vulnerabilities in Chapter 3 and Chapter 4, respectively. Nowadays, network coding has emerged as a promising paradigm for designing wireless routing protocols. Packet mixing operation of network coding based routing brings lots of benefits, such as increasing throughput, reliability enhancement and energy saving. However, this unique operation also makes the network more vulnerable to some new severe security threats that do not exist in the conventional DSR routing protocol. Therefore, in this chapter we focus on the study of security aspects of network coding based routing system. The rest of this chapter is organized as follows. Section 5.1 introduces some background information about network coding system and our motivation. Section 5.2 presents an overall picture of the intra-flow network coding system. Section 5.3 analyzes all possible threats in the entire network coding system. Section 5.4 studies the impacts that are brought by the attackers under various network scenarios. Finally, we conclude this chapter in Section 5.5.

5.1 Introduction

Network coding (NC) has emerged as a promising paradigm for designing wireless routing protocols. In early theoretical study of network coding, researchers proved the benefit of NC for throughput improvement in wired networks. These study results also stimulated wireless communication group to use network coding technique. Nowadays, wireless ad hoc networks become the central application domain of NC. Recent works have also demonstrated the benefits of NC through practical wireless test-beds, such as MORE [74] and COPE [75]. Besides the advantage of increased throughput that is shown in [74-76], NC technique can also provide elegant solution to problems such as reliability enhancement under lossy channel [77] and energy saving [78].

All these benefits of NC are credited to the packet mixing operation, which is very different from conventional routing, where intermediate nodes only forward received packets. The intelligent packet-mixing operation performed by intermediate nodes increases packet diversity in the network, resulting in few redundant transmissions and the better use of network resources. However, this packet-mixing operation also makes the network more vulnerable to a severe security threat known as *packet pollution*. This attack brings devastating impact due to the epidemic effect [79]. Since in NC based routing, intermediate nodes use received packet to generate new coded packet and then broadcast the new one out, a single pollution attack can be quickly amplified and contaminate the entire network.

Most existing works related to NC based routing security only focus on packet pollution attack [79-81]. However, this packet pollution attack is only one of many possible attacks existed in NC based routing protocols. A comprehensive study from

attackers' perspective is still lacking. Therefore we present a general structure to draw insight into all possible attacks that may be launched at different phases of NC based routing. Based on how packets are selected for coding, existing NC based wireless routings can be classified into *intra-flow* and *inter-flow* NC based routings [82]. For intra-flow NC based routing, intermediate nodes only mix packets with the same source and destination. While intermediate nodes in inter-flow systems will mix packets across multiple flows (packets coming from and destined to multiple sources and destinations).

In this chapter, we focus on the study of security aspects of intra-flow NC based routing system. Since we do not know what action an attacker may execute exactly in the next second, each component in a NC system should be equally well defined to prevent attackers from exploiting the weakest part. Therefore, our works and contributions are summarized as follows: 1) We extract the common operations that give an overall picture of the most existing intra-flow NC based routing protocols; 2) We provide a roundup of all possible security threats throughout each operation of NC based wireless routing systems; 3) We simulate attackers' behaviors under different network environments to measure the impact that the attackers may bring to the network; 4) We highlight open questions and road ahead in designing secured NC based routing protocols.

5.2 Intra-flow Network Coding

The general idea of intra-flow network coding is to utilize the broadcast nature with network coding to reduce redundant data transmissions [74, 83]. In intra-flow NC based routing, packets are transmitted in batch with an ID. Each node generates and forwards a batch that is computed by a random linear combination of a set of plain packets. Two batches with the same batch ID indicates that these two batches are composed with the

same set of plain packets. In order to allow destination node to recover the plain packets, the source node keeps computing and broadcasting batches with the same ID until an acknowledgement (ACK) is received from the destination. Intermediate node that has been chosen as a forwarder is responsible to relay batches to the destination. Different from conventional routing protocols where forwarders only relay received packets without changes, in intra-flow NC system, each forwarder stores linearly independent batches it overhears and then forwards a new batch by combining the batches with the same ID stored in its buffer. Once the destination receives enough linear independent batches, it can recover the plain packets that constitute the batch by solving linear equations and reply an ACK to the source, allowing the source node to send the next batch with a new batch ID. Figure 5.1 elaborates the NC based routing process. For example, source **S** generates and broadcasts batches as shown in Figure 5.1. Upon receiving a batch that is linearly independent from already received one, forwarder **F** recreates a new batch and broadcasts it until the destination can decode and issue an ACK. The detail of batch generation operation executed by the source and forwarder is explained in the rest of this section.

Figure 5.1 Intra-flow NC based routing in wireless ad hoc networks.

From the above descriptions we can realize that an intra-flow NC based routing system consists of the following three basic components: *forwarder set selection*, *batch generation* and *acknowledgement replication*. In the rest of this chapter, we discuss each component in detail.

Forwarder set selection: In conventional ad hoc routing protocols, only a single selected next-hop is responsible for forwarding received packets. However, this design is not efficient in front of the lossy wireless channel environment where transmission failure is high, resulting in a lot of retransmissions. In the intra-flow NC system, multiple nodes can receive the broadcast batches. Upon receiving the batch, any selected forwarder can continue to broadcast the batch towards the destination while meeting certain criterion. Many proposed schemes define their own metrics for the forwarder selection, such as expected transmission count (ETX) [84], hop-count [85], expected any path transmission (EAT) [86]. These optimization designs aim to reduce the number of forwarders involved in each forwarding process, so as to reduce congestion and interference.

Batch generation: This mechanism is used to generate coded packet referred to as a *batch*. Before giving the detail, we introduce some notations as listed in Table 5.1. In intra-flow NC system, both source and selected forwarders are responsible to generate and forward batches. Firstly, source node has to divide a serial of plain packets P_i into subsets. Each subset vector, $\vec{G} = [P_1, P_2, \dots, P_n]$, composed by n plain packets will be used to form a batch and broadcast out towards the specified destination. As regulated by the NC system, the source node forms a random linear combination of the subset $\vec{G}$ by equation 5.1 as follows:

$$C = \vec{E} \times \vec{G} \quad (5.1)$$

where $\vec{E} = [e_1, e_2, \dots, e_n]$ is the coding coefficient vector and C is the coded data. Each element in $\vec{E}$ is randomly chosen from a finite field F_q of size q by the source. After that, the source broadcasts the batch consisting of two parts, $(C, \vec{E})$, referred to as *coded data* attached in the payload and *coding coefficient vector* appended in the header. In addition, source node has to assign a *batch ID* that is used to indicate whether two batches are generated by the same set of plain packets. When a selected forwarder receives the batch from the source, it generates a new coded packet by forming random linear combinations of the buffered batch with the same ID. For example, a forwarder buffered a batch, $(C_1, \vec{E}_1)$, that has the same ID as the received batch $(C_2, \vec{E}_2)$, to compute a new batch $(C_3, \vec{E}_3)$, it chooses a random number r then computes $C_3 = r \cdot (C_1 + C_2)$ and $\vec{E}_3 = r \cdot (\vec{E}_1 + \vec{E}_2)$. It can be easily verified that $C_3 = \vec{E}_3 \times G$. Once the destination node receives n linearly independent batches $(C_1, \vec{E}_1), (C_2, \vec{E}_2) \dots (C_n, \vec{E}_n)$, it can retrieve the n plain packets contained in the subset vector $\vec{G}$ by solving a set of linear equations $C_i = \sum_{j=1}^n \vec{E}_j \cdot \vec{G}$.

Table 5.1 Summary of notations for batch generation.

Notations	Descriptions
$\vec{G} = [P_1, P_2, \dots, P_n]$	Subset vector that is composed by plain packets P_1 to P_n
$\vec{E} = [e_1, e_2, \dots, e_n]$	Coding coefficient vector with each element e_i randomly chosen from a finite field F_q of size q
C	Coded data
r	Random number that is also chosen from a finite field F_q of size q

Acknowledgement replication: Once the destination has successfully retrieved all the plain packets via decoding, it initiates an ACK message back to the source using the traditional single path routing with the best quality. All the intermediate nodes presented

in the routing path are supposed to help forwarding the ACK back to the source. Different systems may design their own metrics to measure the routing path quality, which is similar as we described in forwarder set selection part. This ACK message informs the source to send the next batch with a new ID, which is composed by a new set of plain packets.

5.3 Security Threats in Network Coding System

In this section, we analyze all the security vulnerabilities in each component of intra-flow NC based routing system.

5.3.1 Forwarder Set Selection

A critical input for forwarder set selection is the link state information. This information determines the final computation results that are used for forwarder selection, such as the ETX, EAT and hop-count values that have been mentioned in Section 5.2. To obtain the link state information, each node has to monitor the link qualities of its neighbors and periodically broadcasts this information in the entire network. However, an attacker may broadcast false link information in different manners to achieve its attacking purposes.

Black hole attack: The attacker always claims the false but attractive link qualities between itself and its adjacent neighbors. Through this way, the attacker can increase its chance to be selected into forwarder set by other peers. Once the attacker has been selected as a forwarder, it can attract more passing by batches. After that, the attacker can launch more severe attacks, such as some batch generation attacks that we will introduce later. This attack is difficult to prevent, since the information is local to the attacker. *If*

*the network is designed under symmetric network environment, by implementing a cross-checking mechanism executed by each node may alleviate this security threat. For example, node **A**, **B** and **C** are connected mutually, if node **A** finds the link state qualities claimed by node **B** and **C** are not consistent, it can realize the problem.*

Misleading attack: The purpose of this attack is to mislead other peers to include some non-optimal nodes into the forwarder set. In order to conduct this attack, attackers will modify link state information reported by other peers as it is flooded in the network. This attack actually has a similar effect as the black hole attack that makes other peers believe in the existence of a better route to the required destination via an adversary. Nevertheless, different misbehaviors conducted by an attacker require tailored defense schemes. Conventional cryptographic tools, such as hash function, digital signature, can be used to ensure the integrity of the messages. Watchdog mechanism [87] has also been proposed to monitor whether an intermediate node relays a certain message properly. This mechanism is the as the direct observation operation that is introduced in Section 2.3.1.1.

If an attacker purposely injects some false information in preparation for launching an attack, this attack is regarded as an *active attack*. On the contrary, if a node just launches an attack impromptu, this attack is viewed as a *passive attack*. Therefore, we classify both black hole attack and misleading attack into *active attack*.

5.3.2 Batch Generation

Batch generation is the core component of intra-flow NC based routing. Lots of the benefits mentioned in Section 5.1 are credited to this operation. However, this operation also brings new types of attack that only exist in the NC based routing.

Packet pollution attack: This is the most severe attack that has been addressed by the secure NC research group [79-81]. In this attack, the attacker randomly modifies the coding coefficients appended in the head of a received batch. This malicious operation causes the equation 5.1 is not satisfied anymore, which means $C \neq \vec{E} \times \vec{G}$. In other words, the coded data in the batch is not a valid linear combination of plain packets in subset vector $\vec{G}$ with claimed coding coefficient vector $\vec{E} = [e_1, e_2, \dots, e_n]$. As the forwarder combines received batch and stored batch with the same ID to create new batch, this pollution attack will cause epidemic effect. Without a defense mechanism to filter out those polluted batches, if one polluted batch involves in the decoding process, the whole set of retrieved plain packet can be contaminated. Existing defense schemes are generally heavy-weighted, which has been proofed not practical in wireless network [88]. Light-weight approaches for this security issue is still an open problem [80].

Entropy attack: Using coded block which is known as batch will be easy for an attacker to send non-innovative batches that are useless for the destination to decode and retrieve plain packets. Since the attacker only forwards trivial batches that have already existed at the destination, it reduces the diversity of the batches. Therefore, it may take longer time for the destination node to collect adequate linear independent batches for decoding and results in increased delay. This attack is known as entropy attack that only exists in the intra-flow NC based routing system. To the best of our knowledge, few defense schemes address this security threat.

Batch dropping attack: Normally, intra-flow NC routing system should be robust to batch dropping attacks due to the inherent redundancy in broadcast routing manner. However, an over designed forwarder set selection mechanism that aims to reduce the total number of transmissions and interference may cause the side-effect in front of batch

dropping attacks. In addition, under the sparse network environment, if an attacker just placed in a critical point where it is the only gateway between a source and a destination, by simply dropping the received batches, it may degrade the network performance.

5.3.3 Acknowledgement Replication

The acknowledgement (ACK) message issued by a destination is used to inform the source node to start sending the next batch with a new ID. Therefore, a timely and reliable delivery of ACK message is also important to the intra-flow NC based routing systems.

ACK falsification attack: Once the attacker receives or overhears a batch with an ID number, such as ID 1, it impersonates destination to deliver an ACK to falsely claim that all the plain packets in batch with ID 1 has been retrieved. This malicious behavior may cause all plain packets contained in batch with ID 1 to be lost. Since the source node has already moved to deliver the next batch with a new ID, destination may not get enough batches with ID 1 and consequently fail to retrieve all the plain packets in it. To prevent attacker from manipulating multiple identities, the deployment of an authentication mechanism is necessary. Authentication is the first line of security defense against registering any faked ID [89].

ACK dropping attack: If a system is designed to use the most optimized single path for the delivery of ACK message [74], once the attacker lies on the routing path, it can easily drop all passing by ACK packets. Without receiving an ACK packet, the source node will keep sending batches with the same ID continuously and all resources are wasted. Furthermore, an attacker can enhance their chance of being involved in the ACK routing path by executing the black hole attack as we mentioned previously. A bold

design that we think maybe helpful to overcome ACK dropping attacks is that source node counts the number of innovated batches it has transmitted, once it finds the transmitted batches are enough for the destination to retrieve the plain packets, the source node will automatically move to deliver the next batch with a new ID.

In Figure 5.2, we show a complete picture of all the security vulnerabilities of each component that composes the intra-flow NC based routing systems.

Figure 5.2 Structure of existing intra-flow systems and the security vulnerabilities.

5.4 Simulation Results

In this section, we study the impact of attack via NS-2 which is an event driven network simulator [63]. Although each component of intra-flow NC based routing system faces several threats that have addressed in Section 5.3, we focus on those attacks that target on batch generation phase, as these attacks are more specific to NC

environment and more system independent. To make the network model more general to most of the existing intra-flow NC based routing system, we use broadcasting to deliver batches and ACKs, without any forwarder set selection operation. Every node only broadcasts identical batch or ACK packet once, without duplicated transmission. The rest of NC operations are the same as we described in Section 5.2.

5.4.1 Simulator and Simulation Parameters

We simulate a static wireless ad hoc network based on IEEE 802.11 MAC layer protocol. The simulation parameters are listed in Table 5.2.

A Simulated network is specified in a $1000\text{m} \times 1000\text{m}$ square with 50 nodes uniformly distributed in the design region. The number of attackers is set as 10. There is one traffic flow pair as default. The setup of network coding related parameters is recommended by [74]. Coding coefficients are randomly chosen from a finite field of size F_{2^8} , the batch is composed by 8 packets, and the packet size is 512 bytes. Source node adopts a constant rate that broadcasts one batch every 0.25 second. Each node has an interface queue holding up to 50 batches awaiting transmission, and all batches are managed in a drop tail model [90]. All the results are presented with a confidence interval of 95%.

Table 5.2 Simulation parameters.

Default Simulation area	1000m x 1000m
Default number of nodes	50
Default number of attacker	10
Transmission range	250m
Interface queue size	50 batches
Default traffic flow pair	1
Plain packet length	512bytes
Batch size	8 plain packets/batch
Field size of coding coefficient	F_{2^8}
Batch arrival rate	4 batches/s
Simulation time	900s

5.4.2 Attack Patterns

In our simulation, all attacks that target on batch generation phase are studied. Since forwarder set selection operation is not implemented in our model, we only simulate the passive misbehavior of the following attacks:

- *Packet Pollution Attack (PPA)*: Once the attacker receives a batch, it randomly modifies the coding coefficients appended in the head of a received batch, and then broadcasts this modified batch out.
- *Entropy Attack (EA)*: When attacker receives a new batch, it stores this batch in its buffer. After that, once the attacker receives another batch with the same ID as the one stored in its buffer, it broadcasts the one stored in the buffer and drops the received one.
- *Batch Dropping Attack (BDA)*: Attacker drops all received batch packets that it is supposed to help forwarding.

5.4.3 Performance Metrics and Result Analysis

To evaluate the impact brought by the attackers, we use the following metrics:

- *Throughput*: This gives the average number of bits that are correctly decoded by the destinations per second.
- *End-to-end Delay*: This defines the average time a batch takes from the source to the destination and finally being decoded by the destination correctly.

In the rest of this section, we examine the impacts on the network performance by the above discussed attacks under a variety of circumstances.

Effect of number of attacker

In Figure 5.3 we compare the throughput with the presence of three types of attack separately. By varying the number of attackers, we can observe that the throughput drops dramatically with the presence of ten packet pollution attackers (PPA). Once the number of pollution attacker reaches twenty, the throughput almost degrades to the ground. In other words, with the presence of only 10% packet pollution attackers, network performance will degrade more than 60%. This is because the epidemic effect of packet pollution attacks causes all the decoded plain packets are contaminated. For EA and BDA, the attack impact in terms of throughput is not obvious. With the help of broadcast transmission, as long as there are multiple routes between source node and destination node, finally the destination node will be able to decode the batch.

Although the impact of EA and BDA is not obvious in terms of throughput, they do affect on the end-to-end delay as shown in Figure 5.4. Contrary to our intuition, EA causes a longer end-to-end delay compared to the presence of BDA. This is because EA not only reduces the diversity of the encoded batch but also increases the traffic load in the network, in contrast, batch dropping attacks only reduces the diversity of the encoded batch but does not waste the other valuable resources, such as bandwidth. As a result, the end-to-end delay is the longest with the presence of EA. For PPA, the end-to-end delay is as short as always. Since most of batches are polluted by PPA, all those wrongly decoded batches have not been included into the end-to-end delay measurement.

Figure 5.3 Effect of varying number of attackers on throughput.

Figure 5.4 Effect of varying number of attackers on end-to-end delay.

Effect of number of flows

Figure 5.5 presents the impact of attack under different number of traffic flow pairs. From this figure we can observe that even the number of traffic flow pairs increases, the throughput with the presence of (PPA) is still remains very low. This demonstrates the extreme danger of the epidemic effect on the intra-flow network coding based routing. In addition, the throughput degradation due to the presence EA also increases to around 25% once the number of traffic flow pairs reaches four. When the network traffic load becomes heavier, the impact that is brought by the entropy attacks becomes more severe as well. This is because entropy attackers do not help transmitting any innovative batches for the destination nodes to perform decoding instead wasting the network resources. BDA does not have much impact on the throughput and shows almost as good throughput as of attack free. This is benefit from broadcast nature. As long as there is one node which relays batches to the destination, those plain packets will be eventually decoded by the destination.

Figure 5.5 Effect of varying number of flows on throughput.

Effect of simulation area

Size of the simulation area also has the impact on the system throughput. From Figure 5.6 we can observe that when the simulation area becomes larger, the throughput decreases even under attack-free network environment. This is because under a sparse network environment, some batches cannot be received by the destination due to the lacking of forwarders. However, with the presence of any type of attack, the throughput drops much faster compared to the attack-free network environment. When the side length of simulation area reaches 1600m, throughput decreases more than 50% regardless the presences of any type of attack. Since the network becomes sparse, if an attacker in EA or BDA becomes the only gateway between the source and the destination, it will bring severe impact even under broadcasting network environment. For the PPA, the epidemic effect and the sparse network environment turn the network performance down to the worst.

Figure 5.6 Effect of the size of simulation area on throughput.

5.5 Challenges of Secured NC System Design

In this section, we discuss the challenges that are involved in the design of secured NC based routing systems. Based on our previous analysis and evaluation, we point out the following guidelines that are applicable to the design of defense mechanism across different NC systems.

Rule 1: Balancing between forwarder set selection operation and system robustness in front of adversarial network environment.

From above analysis and evaluation results, we can realize that implementing multipath routing with broadcasting manner can greatly enhance the system robustness with the presence of various routing attacks. Benefiting from multipath routing, we can not only avoid those attacks that target the forwarder set selection operation, but also mitigate those packet dropping attacks launched during batch generation and acknowledgment replication phases. Since all components of the NC system are cooperative and they function sequentially, the first step for the design of a secured NC system is to ensure the trustworthiness of selected forwarders. When we design the optimization algorithm for the forwarder set selection, besides contention and interference issues, it is necessary to take security consideration as well, which is lacking in most of the existing NC based routing systems. Over optimization of forwarder set may become the security vulnerability under adversarial network environment.

Rule 2: When utilizing cryptographic tools or any other technique to design a defense mechanism, those practical issues, such as implementation requirements, computational overhead and complexity, must also be borne in mind.

Cryptographic defense mechanisms are always the most common approach to

handle security issues. Not surprisingly, this approach has also been widely proposed to prevent packet pollution attack that brings the most severe negative impact to the network performance. However, in [88] the researchers demonstrated that most of the proposed asymmetric based cryptographic schemes are not practical for NC based routing systems. These schemes impose high communication overhead by placing constraints on the basic NC parameters and high computation overhead by relying on numerous operations over large fields. As a result, they induce a throughput degradation that offset the performance benefit of network coding. Even more, such performance degradation is incurred regardless of the presence of attackers. A symmetric cryptographic scheme named HOMOMAC [91] has been proved to be practical in terms of communication and computational overheads. Unfortunately, this defense mechanism is actually not feasible for implementation on wireless ad-hoc networks. Symmetric cryptographic defense mechanisms not only require a centralized or distributed third party in the network, but also assume the pre-existing shared secret keys between nodes. All these assumptions are invalid in ad hoc networks. As we know, ad-hoc networks are self-organized and have diverse nature of being impromptu, thus they inherently oppose the dependence upon prerequisites. Besides these cryptographic approaches to prevent pollution attacks, some researchers also proposed to tackle this attack by utilizing information theory knowledge [92-94]. These information theory approaches rely on coding redundant information contained in the packets to execute the verification process. Through this way, a receiver can detect the presence of polluted packet on the condition that the adversary has limited bandwidth connected to it. If the attacker has a large bandwidth to the receiver, the verification process may fail and the useful throughput will rapidly degrade to zero. In order to design a practical secured NC based routing

scheme, it should satisfy the following three conditions:

- 1) The performance of the proposed secured NC based routing scheme under attacker free environment should be comparable with the NC routing scheme under the same condition.
- 2) The design of a secure NC based routing scheme must be distributed. So it can be easily implemented on wireless ad hoc networks.
- 3) The performance of the proposed secured NC routing scheme should be relatively robust to the increased number of attackers.

Rule 3: It is necessary to take note of those internal attackers that know how the defense mechanism works. These internal attackers can always explore the loopholes of the defense mechanism and launch some new attacks.

Ad hoc network are self-organized open networks that all participant peers should know how the network is configured. Therefore, it is necessary to assume that all nodes also know how the defense mechanism works. Once some peers inside the network are compromised and become internal attackers, they can easily explore the weakest points and attack the defense mechanism itself. Thus, when we design a secured NC routing scheme, we have to ensure that the internal attackers cannot get much advantage to jeopardize the system even they know how the prevention mechanism works.

5.6 Conclusion

Network coding has emerged as new routing paradigm that can significantly improve the efficiency of network performance by requiring nodes to implement packet mixing before transmitting the packets. Recently, several practical systems have been

proposed to leverage network coding in wireless ad hoc networks. Although the theoretical foundations and benefits of network coding are well understood, security aspects of network coding are still an open problem. The nature of packet mixing and optimization of forwarding nodes selection cause the network coding based routing protocols vulnerable to routing attacks. In this chapter, we conduct a thorough study of the routing attacks under intra-flow network coding system. We reveal that this new routing paradigm is vulnerable to a wide range of attacks at different phases of routing operation. The impacts of different attacks are also varying under different network environments. Although simulation results have shown that the packet pollution attack is the most severe security attack, we have to also alert to other security threats under certain network environments, such as heavy traffic and sparse network. This work is served as a preliminary guide for the future design of secured network coding system.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

The proliferation of mobile devices has led to the growth of wireless mobile ad hoc networks (MANETs). These networks consist of a group of wireless mobile nodes that dynamically exchange data among themselves without the reliance on any centralized administration or fixed base station. Self-organizing characteristic enables MANETs to be widely applied in practice, such as rescue, emergency operations, and battlefield communications, and may bring great revolution in our lives in the near future. However, we have to carefully resolve the security issues before MANETs can be successfully deployed. In this thesis, we focus on providing distributed and practical defense mechanisms to deal with various routing attacks in MANTs. This report can be summarized as follows:

- We thoroughly study all possible ways that the attackers may misbehave during the routing process and analyze the consequences caused by the attackers' misbehaviors. Through this study, we can easily realize that conventional defense techniques relied on cryptographic tools are neither feasible to be implemented in MANETs, nor capable of providing adequate protection in the ad hoc environment where all the nodes are internal entities. There is a need for a solution that offers a lightweight approach to the secured ad hoc routing, while taking into account the security

prerequisites of different application scenarios, providing a flexible approach to the required security-performance balance. Consequently, trust-based routing concept comes into our vision. There are some salient features of trust-based routing concept such as fully distributed design, emulating human being's trust behavior dynamically. All these features make it becomes an ideal candidate for reliable routing in MANETs. Thus, we specifically addressed the key mechanisms of general trust-based routing system, and look into each mechanism to discuss the strengths and weaknesses based on different design requirements. This survey work provides us a preliminary guide to the design of trust-based routing protocols.

- Through previous survey work, we realize that if the attacker can achieve route invasion, it will bring most severe impact on the network performance. A typical route invasion attack is called the black hole attack. If a node claims that it has the shortest path to the destination, but actually it does not have the proper routing information, this attacker is known as active black hole attacker. Active black hole attackers can attract lots of data packets and drops them silently. On the contrary, if the node behaves properly during route discovery but discards all passing-by data packets when it is selected as the final routing path for data transmission, it is known as passive black hole. With the presence of only 10% of passive black hole attackers, packet delivery ratio of the network will drastically decrease around 40%. Compared to network with passive black hole nodes, network performance with active black hole nodes is even worse. Therefore, we proposed a self-checking trust scheme that targets on the active black hole attacks. This self-checking trust scheme requires every node to exchange neighbor information before route discovery, and then uses previously collected neighboring information to verify each received route reply. As

the attackers do not know who will be the requested destination in next route request message, they have no idea what neighbor information they need to fake in order to avoid being caught. If they randomly add many faked neighbors into broadcast Hello messages, they can be easily identified. If certain node refuses to exchange neighbor information, definitely it will be caught if it behaves as an active black hole attacker in next second. If attackers provide faked neighbor information after they know the requested destination, liar-checking operation will function. By utilizing previously collected neighboring information we can greatly increase the robustness of our prevention system. Extensive simulations confirmed the effectiveness of this self-checking trust scheme. We believe this distributed, simple and effective defense solution should be more suitable for practical MANET operation scenarios.

- After solving the active black hole attack, we turn our attention to the passive black hole attack. Compared to the active black hole, passive black hole is more covert to be identified. Since the passive black hole node does not inject any false information during rout discovery, but really presented in the routing path. Hence, we propose an evolutionary self-cooperative trust scheme (ETS) to defend against various passive black hole attacks. This ETS emulated human cognition promotes network scalability and ensure reliable routing in MANETs via three aspects. Firstly, self-detection mechanism utilizes mobility to perform trust analysis. The idea of self-detection essentially resembles human behavior for which the most secure way to obtain information from a person is by approaching him/her directly. Since every node is mobile in MANETs, it is possible for every two nodes to become direct neighbors. Once a source or intermediate node realizes a past destination becomes a direct neighbor, it sends an investigation request immediately to check how many

data packet the destination received along a specified routing path in the past. Based on the investigation reply from the destination, requested node can judge whether there are any hidden malicious nodes between itself and the destination node in the specified routing path. Self-detection provides a baseline that ensures the accuracy of trust evaluation. Secondly, cooperative detection mechanism can further improve the performance and enhance the robustness of this trust system. On the one hand, cooperative detection speeds up the trust evaluation process and mitigates trust information shortage issue. On the other hand, voting concept implied in the cooperative detection ensures that as long as the majority of the nodes inside the network are honest, forming a malicious collective will not increase the trust value of malicious peers enough in order for them to have impact on the network. Thirdly, evolutionary operation of ETS scheme enables nodes to explore diverse form of networks, to alleviate the impact of false information provided by the dishonest entities and correct their cognition by themselves. As long as the majority of nodes inside the network are honest, this ETS can improve the PDR more than 90% as compared to basic DSR regardless the types of passive routing attacks.

- Since network coding has emerged as new routing paradigm, we also conduct a detailed analysis of various routing security threats that attack on different components of intra-flow network coding based routing schemes. At the same time, we provide an extensive simulation study of the impacts that are brought by various attackers. We reveal that packet pollution is the most severe security attack under network environment. However, the impacts of different attacks are also varying under different network environment, such as heavy traffic and sparse network scenarios.

6.2 Recommendation for Future Work

Since network coding has emerged as a promising routing paradigm for designing wireless routing protocols, our future work will focus on the routing security aspects under network coding environment. We suggest the following topics for our future works:

- In intra-flow network coding system nodes only mix packets belonging to the same flow (packets with the same source and destination). Differently, nodes in inter-flow network coding systems will mix packets across multiple flows (packets may coming from and destined to multiple sources and destinations). In intra-flow network coding environment, routing is implemented in broadcast manner and coding may occur at all nodes. In contrast, inter-flow network coding systems are generally designed on top of existing routing protocols, and coding only occurs at nodes located at the intersection of flows. At last, for intra-flow network coding, packet mixing is performed through linear network coding, while the packet mixing in inter-flow network coding is through XOR operation. All these differences between intra-flow and inter-flow network coding networks make the security threats also various in facing the two types of network coding environments. To give a more comprehensive understanding of routing security issues under network environment, we will also analysis and simulate those routing attacks existed under inter-flow network environment in the future.
- Most existing works related to network coding security focus on pollution attacks. Quite a few defense mechanisms that use special-crafted digital signatures and hash function with homomorphic properties [95-97] are designed to allow forwarders to verify the integrity of coded packets. While these elegant approaches come from

theoretical perspective, they have been proved inefficient when they are applied in practice in wireless ad hoc networks due to high computational overhead [88]. Some non-cryptographic and light weight defense schemes have also been proposed [92, 93] to defense pollution attacks. However, these solutions either provide only partial solution by detecting the attacks without any response mechanism [92], or add redundant data at the source, resulting in throughput degradation proportionally to the bandwidth available to the attacker [93]. Therefore, a practical and efficient defense mechanism is highly demanded to address the security issues in network coding environment. We believe trust concept can still contribute in achieving security for network coding based routing systems.

Author's Publications

Journal Papers

1. R. J. Cai, X. J. Li and P. H. J. Chong, "A Novel Self-Checking Ad Hoc Routing Scheme Against Active Black Hole Attacks" *Security and Communication Networks*, vol. 9, pp. 943-957, 2016.

Conference Papers

1. R. J. Cai, G. G. M. Nawaz Ali, C. Y. Aung, M. Zhao and P. H. J. Chong "Simulation Study of Routing Attacks Under Network Coding Environment", IEEE ICC'16, Kuala Lumpur, Malaysia, May 23-27, 2016.
2. R. J. Cai and P. H. J. Chong "A Self-Cooperative Trust Scheme for Message Propagation in VANETs" in *Proc. of 14th ITS Asia Pacific Forum*, Nanjing, China, April, 2015.
3. R. J. Cai and P. H. J. Chong, "A Neighborhood Connectivity-based Trust Scheme to Identify Active Black Hole Attacks" in *Proc. of IEEE International Conference on Communication System*, Macau, November, 2014.
4. R. J. Cai, P. H. J. Chong and C. Y. Aung "Poster: Trust-based Routing With Neighborhood Connectivity to Prevent Single and Colluded Active Black Hole" in *Proc. of International Conference on Communication and Networking in China (CHINACOM)*, Maoming, August, 2014.
5. G. H. Eng, R. J. Cai and P. H. J. Chong "Markov analysis of modulation & coding schemes in OFDMA networks" in *Proc. of IEEE International Conference on Communication System*, Macau, November, 2014.

6. C. Y. Aung, P. H. J. Chong and R. J. Cai “Hybrid Opportunistic Routing in Highly Dynamic MANET” in *Proc. of Computer Communication and Networks (ICCCN)*, Shanghai, August, 2014.

Working Papers

1. R. J. Cai, W. C. W. Tan and P. H. J. Chong, “An Overview of Trust-based Routing Design Issues in Mobile Ad Hoc Networks”. Paper in progress.
2. R. J. Cai, X. J. Li and P. H. J. Chong, “An Evolutionary Self-Cooperative Trust Scheme Against Routing Disruptions in MANETs”. Paper in progress.

Bibliography

- [1] C. E. Perkins, E. M. Belding-Royer, and S. R. Das, Ad hoc on demand distance vector (AODV) Routing: *IETF RFC 3561*, July 2003.
- [2] D. B. Johnson, D. A. Maltz, Y. C. Hu, and J. G. Jetcheva, The dynamic source routing protocol for mobile ad hoc networks (DSR): *IETF RFC 4728*, February 2007.
- [3] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," *SIGCOMM Comput. Commun. Rev.*, vol. 24, pp. 234-244, 1994.
- [4] T. Clausen and P. Jacquet, Optimized link state routing protocol (OLSR): *IETF RFC 3626*, October 2003.
- [5] A. Pirzada and C. McDonald, "Secure routing protocols for mobile ad hoc wireless networks," *Advanced Wired and Wireless Networks*, vol. 26, pp. 57-80, 2005.
- [6] Y. C. Hu, A. Perrig, and D. B. Johnson, "Ariadne: a secure on-demand routing protocol for ad hoc networks," *Wireless Networks*, vol. 11, pp. 21-38, 2005.
- [7] M. G. Zapata and N. Asokan, "Securing ad hoc routing protocols," in *Proc. of the ACM Workshop on Wireless Security*, Atlanta, United States, 2002, pp. 1-10.
- [8] K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer, "A secure routing protocol for ad hoc networks," in *Proc. of 10th IEEE International Conference on Network Protocols*, 2002, pp. 78-87.

- [9] C. Castelluccia and G. Montenegro, "Protecting AODV against impersonation attacks," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 6, pp. 108-109, 2002.
- [10] M. M. G. Pradip M. Jawandhiya , M.S.Ali , J.S. Deshpande, "A survey of mobile ad hoc network attacks," *International Journal of Engineering Science and Technology*, vol. 2(9), 2010.
- [11] B. Wu, J. Chen, J. Wu, and M. Cardei, "A survey of attacks and countermeasures in mobile ad hoc Networks," in *Wireless Network Security*, pp. 103-135, 2007.
- [12] N. Peng and S. Kun, "How to misuse AODV: a case study of insider attacks against mobile ad-hoc routing protocols," *Ad Hoc Networks*, vol. 3, pp. 795-819, 2005.
- [13] J. H. Cho, A. Swami, and I. R. Chen, "A survey on trust management for mobile ad hoc networks," *IEEE Communications Surveys & Tutorials*, vol. 13, pp. 562-583, 2011.
- [14] K. Hoffman, D. Zage, and C. N. Rotaru, "A survey of attack and defense techniques for reputation systems," *ACM Computing Surveys*, vol. 42, pp. 1-31, 2009.
- [15] Y. Han, S. Zhiqi, M. Chunyan, C. Leung, and D. Niyato, "A survey of trust and reputation management systems in wireless communications," *Proceedings of the IEEE*, vol. 98, pp. 1755-1772, 2010.
- [16] A. Nadeem and M. P. Howarth, "A survey of MANET intrusion detection and prevention approaches for network layer attacks," *IEEE Communications Surveys & Tutorials*, vol. 15, pp. 2027-2045, 2013.
- [17] J. H. Cho and I. R. Chen, "On the tradeoff between altruism and selfishness in MANET trust management," *Ad Hoc Networks*, vol. 11, pp. 2217-2234, 2013.

- [18] J. Jiang, G. Han, F. Wang, L. Shu, and M. Guizani, "An efficient distributed trust model for wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, pp. 1228-1237, 2015.
- [19] K. Osathanunkul and Z. Ning, "A countermeasure to black hole attacks in mobile ad hoc networks," in *Proc. of IEEE International Conference on Networking, Sensing and Control (ICNSC)*, 2011, pp. 508-513.
- [20] Y. Guo and S. Perreau, "Detect DDoS flooding attacks in mobile ad hoc networks," *International Journal of Security Networks.*, vol. 5, pp. 259-269, 2010.
- [21] J. Luo, M. Fan, and D. Ye, "Black hole attack prevention based on authentication mechanism," in *Proc. of 11th IEEE Singapore International Conference on Communication Systems (ICCS)*, 2008, pp. 173-177.
- [22] Y. Wu, Y. Zhao, M. Riguidel, G. Wang, and P. Yi, "Security and trust management in opportunistic networks: a survey," *Security and Communication Networks*, vol. 8, pp. 1812-1827, 2015.
- [23] J. H. Cho, K. S. Chan, and I. R. Chen, "Composite trust-based public key management in mobile ad hoc networks," in *Proc. of the 28th Annual ACM Symposium on Applied Computing*, Coimbra, Portugal, 2013.
- [24] K. Govindan and P. Mohapatra, "Trust computations and trust dynamics in mobile adhoc networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 14, pp. 279-298, 2012.
- [25] R. Floriano De, "Trust-based SAODV protocol with intrusion detection and incentive cooperation in MANETs," *International Journal of Interdisciplinary Telecommunications and Networking (IJITN)*, vol. 1, pp. 54-70, 2009.

- [26] N. L. C. D. Harrison Mcknight "The meanings of trust," University of Minnesota, Technical Report TR 94-04, 1996.
- [27] Y. Wei, S. Yan, and K. J. R. Liu, "HADOF: defense against routing disruptions in mobile ad hoc networks," in *Proc. of the IEEE INFOCOM 2005*, pp. 1252-1261.
- [28] K. RahimiZadeh and P. Kabiri, "Trust-based routing method using a mobility-based clustering approach in mobile ad hoc networks," *Security and Communication Networks*, vol. 7, pp. 1746-1763, 2014.
- [29] D. Djenouri and N. Badache, "Struggling against selfishness and black hole attacks in MANETs," *Wireless Communications and Mobile Computing*, vol. 8, pp. 689-704, 2008.
- [30] W. Yu and K. J. Ray Liu, "Attack-resistant cooperation stimulation in autonomous Ad Hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, pp. 2260-2271, 2005.
- [31] A. A. Pirzada, C. McDonald, and A. Datta, "Performance comparison of trust-based reactive routing protocols," *IEEE Transactions on Mobile Computing*, vol. 5, pp. 695-710, 2006.
- [32] S. Buchegger and J. Y. Le Boudec, "Self-policing mobile ad hoc networks by reputation systems," *IEEE Communications Magazine*, vol. 43, pp. 101-107, 2005.
- [33] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *Proc. of the 6th annual international conference on Mobile computing and networking*, Boston, Massachusetts, USA, 2000, pp. 255-265.

- [34] B. Yang, Y. Ryo, and T. Yoshiaki, "Historical evidence based trust management strategy against black hole attacks in MANET," in *Proc. of 14th International Conference on Advanced Communication Technology (ICACT)*, 2012 pp. 394-399.
- [35] T. Ghosh and N. Pissinou, "Towards designing a trusted routing solution in mobile ad hoc networks," *Mobile Networks and Applications*, vol. 10, pp. 985-995, 2005.
- [36] W. Gong, Z. You, D. Chen, X. Zhao, M. Gu, and K. Y. Lam, "Trust based routing for misbehavior detection in Ad hoc networks," *Journal of Networks*, vol. 5, pp. 551-558, 2010.
- [37] W. Li, J. Parker, and A. Joshi, "Security through collaboration and trust in MANETs," *Mobile Network Application*, vol. 17, pp. 342-352, 2012.
- [38] H. Deng, W. Li, and D. P. Agrawal, "Routing security in wireless ad hoc networks," *IEEE Communications Magazine*, vol. 40, pp. 70-75, 2002.
- [39] H. Weerasinghe and H. Fu, "Preventing cooperative black hole attacks in mobile ad hoc networks: Simulation implementation and evaluation," in *Proc. of the Future Generation Communication and Networking (FGCN)*, 2007, pp. 362-367.
- [40] P. C. Tsou, J. M. Chang, H. C. Chao, and J. L. Chen, "Developing a BDSR scheme to avoid black hole attack based on proactive and reactive architecture in MANETs," in *Proc. of the 13th International Conference on Advanced Communication Technology (ICACT)*, Seoul, Korea, 2011, pp. 1-5.
- [41] R. J. Cai and P. H. J. Chong, "A neighborhood connectivity-based trust scheme to identify active black hole attacks," in *Proc. of IEEE International Conference on Communication Systems (ICCS)* 2014, pp. 543-548.

- [42] P. Michiardi and R. Molva, "CORE: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks," in *Proc. of Joint Working Conference on Communications and Multimedia Security*, Norwell, MA, USA, 2002, pp. 107-120.
- [43] C. Zouridaki, B. L. Mark, M. Hejmo, and R. K. Thomas, "Hermes: A quantitative trust establishment framework for reliable data packet delivery in MANETs," *Journal of Computer Security.*, vol. 15, pp. 30-38, 2007.
- [44] C. T. Nguyen, O. Camp, and S. Loiseau, "A Bayesian network based trust model for improving collaboration in mobile ad hoc networks," in *Proc. of IEEE International Conference on Research, Innovation and Vision for the Future*, Piscataway, NJ, USA, 2007, pp. 144-151.
- [45] H. Chen, H. Wu, X. Zhou, and C. Gao, "Agent-based trust model in wireless sensor networks," in *Proc. of 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing* Piscataway, NJ, USA, 2007, pp. 119-124.
- [46] S. Yan Lindsay, Y. Wei, H. Zhu, and K. J. R. Liu, "Information theoretic framework of trust modeling and evaluation for ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, pp. 305-317, 2006.
- [47] Y. Wang and M. P. Singh, "Formal trust model for multiagent systems," in *Proc. of 20th International Joint Conference on Artificial Intelligence*, Hyderabad, India, 2007, pp. 1551-1556.
- [48] X. Ding and H. Luo, "Trust evaluation based reliable routing in wireless mesh network," in *Proc. of 3rd International Conference on Wireless Communications, Networking, and Mobile Computing*, Piscataway, NJ, USA, 2007, pp. 294-230.

- [49] J. Li, R. Li, and J. Kato, "Future trust management framework for mobile ad hoc networks," *IEEE Communications Magazine*, vol. 46, pp. 108-114, 2008.
- [50] K. D. Meka, M. Virendra, and S. Upadhyaya, "Trust Based Routing Decisions in Mobile Ad-hoc Networks," in *Proc. of the Workshop on Secure Knowledge Management*, 2006.
- [51] P. B. Velloso, R. P. Laufer, D. D. O. Cunha, O. C. M. B. Duarte, and G. Pujolle, "Trust management in mobile ad hoc networks using a scalable maturity-based model," *IEEE Transactions on Network and Service Management*, vol. 7, pp. 172-185, 2010.
- [52] P. Gera, K. Garg, and M. Misra, "Trust based multi path DSR protocol," in *Proc. of Availability, Reliability, and Security*, 2010.
- [53] I. R. Chen, J. Guo, F. Bao, and J. H. Cho, "Trust management in mobile ad hoc networks for bias minimization and application performance maximization," *Ad Hoc Networks*, vol. 19, pp. 59-74, 2014.
- [54] F. D. Rango, "Improving SAODV protocol with trust levels management, IDM and incentive cooperation in MANET," in *Proc. of Wireless Telecommunications Symposium*, 2009, pp. 1-8.
- [55] F. H. Tseng, L. D. Chou, and H. C. Chao, "A survey of black hole attacks in wireless mobile ad hoc networks," *Human-centric Computing and Information Sciences*, vol. 1, pp. 1-16, 2011.
- [56] V. N. Padmanabhan and D. R. Simon, "Secure traceroute to detect faulty or malicious routing," *ACM SIGCOMM Computer Communication Review*, vol. 33, pp. 77-82, 2003.

- [57] S. Djahel, F. Nait-Abdesselam, and A. Khokhar, "An acknowledgment-based scheme to defend against cooperative black hole attacks in optimized link state Routing Protocol," in *Proc. of IEEE International Conference on Communications (ICC)*, 2008, pp. 2780-2785.
- [58] M. Al-Shurman, S.-M. Yoo, and S. Park, "Black hole attack in mobile Ad Hoc networks," in *Proc. of the 42nd annual Southeast regional conference*, Huntsville, Alabama, 2004, pp. 96-97.
- [59] S. Misra, P. V. Krishna, A. Bhiwal, A. Chawla, B. Wolfinger, and C. Lee, "A learning automata-based fault-tolerant routing algorithm for mobile ad hoc networks," *The Journal of Supercomputing*, vol. 62, pp. 4-23, 2012.
- [60] P. Narula, S. K. Dhurandher, S. Misra, and I. Woungang, "Security in mobile ad-hoc networks using soft encryption and trust-based multi-path routing," *Computer Communications* vol. 31, pp. 760-769, 2008.
- [61] B. J. Oommen and S. Misra, "Fault-tolerant routing in adversarial mobile ad hoc networks: an efficient route estimation scheme for non-stationary environments," *Telecommunication Systems*, vol. 44, pp. 159-169, 2010.
- [62] C. Bettstetter, G. Resta, and P. Santi, "The node distribution of the random waypoint mobility model for wireless ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 2, pp. 257-269, 2003.
- [63] K. Fall and K. Varadhan, "ns notes and documentation," *available from* <http://www-mash.cs.berkeley.edu/ns/>, 1999.
- [64] M. Yu, M. Zhou, and W. Su, "A secure routing protocol against byzantine attacks for MANETs in adversarial environments," *IEEE Transactions on Mobile Computing*, vol. 58, pp. 449-460, 2009.

- [65] M. Ge, K. Y. Lam, D. Gollmann, S. L. Chung, C. C. Chang, and J. B. Li, "A robust certification service for highly dynamic MANET in emergency tasks," *International Journal of Communication Systems*, vol. 22, pp. 1177-1197, 2009.
- [66] J. Luo, J. P. Hubaux, and P. T. Eugster, "DICTATE: Distributed certification authority with probabilistic freshness for ad hoc networks," *IEEE Transactions on Dependable and Secure Computing.*, vol. 2, pp. 311-323, 2005.
- [67] J. Al-Karaki and A. Kamal, "Stimulating node cooperation in mobile ad hoc networks," *Wireless Personal Communications*, vol. 44, pp. 219-39, 2008.
- [68] S. Zhong, J. Chen, and Y. R. Yang, "Sprite: a simple, cheat-proof, credit-based system for mobile ad-hoc networks," in *Proc. of IEEE INFOCOM 2003*, pp. 1987-1997
- [69] S. Buchegger and J. Y. L. Boudec, "Performance analysis of the CONFIDANT protocol " in *Proc. of ACM MobiHoc '02*, New York, USA, 2002, pp. 226-236.
- [70] T. Chen, L. Zhu, F. Wu, and S. Zhong, "Stimulating cooperation in vehicular ad hoc networks: A coalitional game theoretic approach," *IEEE Transactions on Vehicular Technology* vol. 60, pp. 566-579, 2011.
- [71] L. Butty and J. P. Hubaux, "Stimulating cooperation in self-organizing mobile ad hoc networks," *Mobile Network and Applications.*, vol. 8, pp. 579-592, 2003.
- [72] A. JØSANG, "A logic for uncertain probabilities," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 09, pp. 279-311, 2001.
- [73] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, pp. 30-37, 2009.

- [74] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading structure for randomness in wireless opportunistic routing," *SIGCOMM Comput. Commun. Rev.*, vol. 37, pp. 169-180, 2007.
- [75] S. Katti, D. Katabi, W. Hu, H. Rahul, and M. Medard, "The Importance of being opportunistic: practical network coding for wireless environments," in *Proc. of Allerton Annual Conference on Communication, Control and Computing*, 2005.
- [76] J. Le, J. C. S. Lui, and D. M. Chiu, "DCAR: Distributed coding-aware routing in wireless networks," *IEEE Transactions on Mobile Computing*, vol. 9, pp. 596-608, 2010.
- [77] F. Ye, S. Roy, and H. Wang, "Efficient data dissemination in vehicular ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 30, pp. 769-779, 2012.
- [78] Y. Wu, P. A. Chou, and S. Y. Kung, "Minimum-energy multicast in mobile ad hoc networks using network coding," *IEEE Transactions on Communications*, vol. 53, pp. 1906-1918, 2005.
- [79] F. Oggier and H. Fathi, "An Authentication code against pollution attacks in network coding," *IEEE/ACM Transactions on Network*, vol. 19, pp. 1587-1596, 2011.
- [80] J. Dong, R. Curtmola, and C. Nita-Rotaru, "Practical defenses against pollution attacks in intra-flow network coding for wireless mesh networks," in *Proc. of the second ACM conference on Wireless network security*, Zurich, Switzerland, 2009, pp. 111-122.

- [81] Z. Yu, Y. Wei, B. Ramkumar, and G. Yong, "An efficient scheme for securing XOR network coding against pollution attacks," in *Proc. of IEEE INFOCOM 2009*, Rio de Janeiro, pp. 406-414.
- [82] M. A. Iqbal, B. Dai, B. Huang, A. Hassan, and S. Yu, "Survey of network coding-aware routing protocols in wireless networks," *Journal of Network and Computer Applications*, vol. 34, pp. 1956-1970, 2011.
- [83] L. Peng, G. Song, Y. Shui, and A. V. Vasilakos, "Reliable multicast with pipelined network coding using opportunistic feeding and routing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, pp. 3264-3273, 2014.
- [84] T. Meng, F. Wu, Z. Yang, G. Chen, and A. V. Vasilakos, "Spatial reusability-aware routing in multi-hop wireless networks," *IEEE Transactions on Computers*, vol. 65, pp. 244-255, 2016.
- [85] M. S. Nassr, J. Jun, S. J. Eidenbenz, A. A. Hansson, and A. M. Mielke, "Scalable and reliable sensor network routing: Performance study from field deployment," in *Proc. of IEEE INFOCOM 2007*, Anchorage, AK, pp. 670-678.
- [86] Z. Zhong, J. Wang, S. Nelakuditi, and G. H. Lu, "On selection of candidates for opportunistic anypath forwarding," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 10, pp. 1-2, 2006.
- [87] X. Lou, H. Yao, C. W. Tan, and J. Wang, "VANDER: Efficient cooperative watchdog monitoring for lossy wireless network coding," *IEEE Transactions on Vehicular Technology*, vol. 64, pp. 702-713, 2015.
- [88] A. Newell, J. Dong, and C. Nita-Rotaru, "On the practicality of cryptographic defences against pollution attacks in wireless network coding," *ACM Comput. Surv.*, vol. 45, pp. 1-26, 2013.

- [89] A. Attat, H. Tang, A. V. Vasilakos, F. R. Yu, and V. Leung, "A survey of security challenges in cognitive radio networks: Solutions and future research directions," *Proceedings of the IEEE*, vol. 100, pp. 3172-3186, 2012.
- [90] S. R. Das, C. E. Perkins, and E. M. Royer, "Performance comparison of two on-demand routing protocols for ad hoc networks," in *Proc. of the IEEE INFOCOM 2000*, Tel Aviv, pp. 3-12.
- [91] S. Agrawal and D. Boneh, "Homomorphic MACs: MAC-based integrity for network coding," *Applied Cryptography and Network Security*, vol. 5536, pp. 292-305, 2009.
- [92] H. Tracey, B. Leong, R. Koetter, M. Medard, M. Effros, and D. R. Karger, "Byzantine modification detection in multicast networks using randomized network coding," in *Proc. of International Symposium on Information Theory*, 2004, pp. 144-152.
- [93] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, and M. Medard, "Resilient network coding in the presence of byzantine adversaries," in *Proc. of 26th IEEE International Conference on Computer Communications, INFOCOM 2007*, Barcelona, pp. 616-624.
- [94] R. Koetter and F. R. Kschischang, "Coding for errors and erasures in random network coding," *IEEE Transactions on Information Theory*, vol. 54, pp. 3579-3591, 2008.
- [95] D. Charles, K. Jain, and K. Lauter, "Signatures for network coding," in *Proc. of 40th Annual Conference on Information Sciences and Systems*, 2006.
- [96] C. Gkantsidis and P. R. Rodriguez, "Cooperative security for network coding file distribution," in *Proc. of INFOCOM 2006*, Baelona, Spain, pp. 1-13.

- [97] F. Zhao, T. Kalker, M. Medard, and K. J. Han, "Signatures for content distribution with network coding," in *Proc. of IEEE International Symposium on Information Theory (ISIT)*, 2007.