Towards security and privacy for emerging wireless networks

Pham, Thi Ngoc Diep

2016

Pham, T. N. D. (2016). Towards security and privacy for emerging wireless networks.
Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/69407>

<https://doi.org/10.32657/10356/69407>

NANYANG
TECHNOLOGICAL
UNIVERSITY

**TOWARDS SECURITY AND PRIVACY
FOR EMERGING WIRELESS NETWORKS**

PHAM THI NGOC DIEP
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING
2016

**TOWARDS SECURITY AND PRIVACY
FOR EMERGING WIRELESS NETWORKS**

PHAM THI NGOC DIEP

School of Computer Science and Engineering

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

2016

Acknowledgements

First I would like to express my sincere gratitude to my supervisor, Associate Professor Yeo Chai Kiat, for the continuous guidance and encouragement during the last four years of my Ph.D study. Thanks to her immense knowledge and patient instruction, I could learn a lot and get motivated in my research. Moreover, she offers various opportunities for me to broaden my academic experiences. This dissertation would not be possible without her precious support.

I would like to take this opportunity to thank my labmates in Center for Multimedia and Network Technology (CeMNet), including Bong Zoebir, Xia Yang, Mao Yuanyuan, Goh Wei-han, Yang Yiqun, Chen Zhaomin, Chen Weiling, Yean Seanglidet. I also want to give special thanks to my friend Hou Yuxiao in Parallel Distributed Computing Center (PDCC). They have always supported me with their great advices, companionship and encouragement.

My work cannot progress smoothly without the assistance of the lab officers. I would like to thank Ms. Chua Poo Hua, Ms. Siom Siew Ling and Mr. Wong Jun Hao for their hard work to maintain the lab and their help with computing resource issues.

Last but not least, I am thankful to my parents and my younger sister for their continuous support and endless love. This thesis is dedicated to them.

Contents

Acknowledgements	i
List of Figures	vii
List of Tables	x
List of Abbreviations	xi
Abstract	xiii
1 Introduction	1
1.1 Background and Research Methodology	1
1.2 Motivation and Scope	3
1.2.1 Wormhole attack in DTN	3
1.2.2 Blackhole and greyhole attacks in DTN	6
1.2.3 Flooding attack in DTN	7
1.2.4 Trust and privacy trade-off in VANET	9
1.3 Major Contributions	11
1.3.1 Detecting wormhole attack in DTN	11
1.3.2 Detecting blackhole and greyhole attacks in DTN	12
1.3.3 Detecting flooding attack in DTN	13
1.3.4 Balancing trust management and privacy in VANET	14
1.4 Thesis Organization	15
2 Literature Review	17
2.1 Wormhole Attack	17

2.1.1	Detecting wormhole attack in MANET	17
2.1.2	Detecting wormhole in DTN	22
2.2	Blackhole and Greyhole Attack	23
2.2.1	Methods based on delivery predictability records	24
2.2.2	Methods based on forwarding receipts and encounter records . .	25
2.3	Flooding Attack	28
2.3.1	Detecting flooding in wired network and MANET	28
2.3.2	Detecting flooding in DTN	29
2.4	Trust Management and Privacy Preservation in VANET	31
2.4.1	Privacy preservation	31
2.4.2	Trust management	36
2.4.3	Trust management in privacy context	41
3	Detecting and Localizing Wormhole Attack in DTN	43
3.1	System Model	44
3.2	SNDW	45
3.2.1	Detecting wormhole	45
3.2.2	Localizing wormhole	47
3.2.3	Discussion on statistical approaches to detect wormhole	48
3.3	Performance Evaluation	50
3.3.1	Comparing SNDW with prohibited-topology method	51
3.3.2	Varying network parameters	52
3.3.3	Varying method parameters	54
3.4	Conclusion	56
4	Detecting Blackhole and Greyhole Attacks in DTN	57
4.1	System Model	58
4.1.1	Mobile node model	58
4.1.2	Individual packet-dropping attack model	60
4.2	Detecting Individual Attackers	61
4.2.1	ER manipulation misbehavior	62
4.2.2	Dropping misbehavior	64

4.3	Detecting Collusion Attackers	65
4.3.1	Collusion packet-dropping attack model	66
4.3.2	Analysis of collusion behavior	69
4.3.3	Detection of collusion	72
4.3.4	Special case: collusion behavior-switching attack	74
4.4	Performance Evaluation	75
4.4.1	Simulation setup	75
4.4.2	Evaluating the detection performance of SDBG	76
4.4.3	Comparing with other schemes	82
4.4.4	Storage overhead	87
4.4.5	Mitigating collusion behavior-switching attack	90
4.5	Conclusion	92
5	Detecting Flooding Attack in DTN	93
5.1	Overview	94
5.1.1	Background and motivation	94
5.1.2	Defense lines against flooding attack	95
5.1.3	System model	96
5.1.4	Basic ideas	97
5.2	Misbehavior detection scheme	98
5.2.1	Message model	98
5.2.2	ER model	101
5.2.3	Detecting ER manipulation	103
5.2.4	Detecting message count manipulation	107
5.2.5	Detecting replica count manipulation	109
5.2.6	How ER window mitigates flooding attack	111
5.3	Fairness Forwarding and Queueing Policy	113
5.3.1	Updating the forwarding priority	113
5.3.2	Fairness policy	115
5.3.3	Integrating fair policy with existing routing protocols	117
5.4	Performance Evaluation	117
5.4.1	Simulation setup	117

5.4.2	Detecting ER manipulation misbehavior	118
5.4.3	Detecting message flooding	119
5.4.4	Detecting replica flooding	124
5.4.5	Choosing window size	125
5.5	Conclusion	127
6	Balancing Trust and Privacy in VANET	128
6.1	Overview	129
6.1.1	Network model	129
6.1.2	Problem statement	130
6.1.3	Basic idea of linkability scheme	131
6.2	Preliminary	134
6.2.1	PSI	134
6.2.2	PSI-CA	136
6.3	ALRS	138
6.3.1	Identity linkability establishment protocol	139
6.3.2	Identity recognition protocol	140
6.3.3	Trust recognition protocol	142
6.3.4	Security analysis	143
6.4	ATMS	145
6.4.1	Application scenario: LDW	146
6.4.2	Objective evaluation of event	148
6.4.3	Subjective evaluation of event	150
6.4.4	Update of peers' reputation	154
6.4.5	Balancing trust and privacy	155
6.5	Performance Evaluation	156
6.5.1	Setup	156
6.5.2	Results	157
6.5.3	Overhead	159
6.6	Conclusion	160
7	Conclusions and Future Work	162

7.1	The Enhancement of Security and Privacy for DTN and VANET	163
7.1.1	Security against wormhole attack	163
7.1.2	Security against blackhole and greyhole attacks	164
7.1.3	Security against flooding attack	165
7.1.4	Security and privacy balance	165
7.2	Future Research Directions	166
Bibliography		168
Author's Publications		178

List of Figures

1.1	Wormhole attack.	4
1.2	Examples of attacks and detection schemes. n_1, n_2, n_3 are normal nodes. m and m' are malicious nodes.	8
2.1	Hypothetical VS real distribution of network parameters.	20
2.2	Example of statistical change in WSN.	21
2.3	Prohbitted topology due to wormhole.	23
2.4	Blackhole defense using reputation and forwarding receipt.	25
3.1	(a) Distribution of helper nodes. (b) Movement of helper nodes.	45
3.2	Example of neighbor count statistic.	46
3.3	Possible wormhole location corresponding to wormhole declaration by: (a) four helpers, (b) two helpers, (c) one helper	48
3.4	Distribution of neighbor counts in DTN with and without wormhole.	49
3.5	Statistical change of neighbor counts in DTN.	50
3.6	Performance of SNDW under varying transmission range.	53
3.7	Performance of SNDW under varying number of nodes.	53
3.8	Performance of SNDW under varying method parameters: (a) Detection time. (b) Detection accuracy. (c) False positive.	55
4.1	ER manipulation strategies of attackers.	61
4.2	Detecting inconsistent assignment of ER sequence number.	63
4.3	Encounter records frequency between node pairs.	69
4.4	Average number of sent messages between node pairs.	70
4.5	FXS metric of forged ER vs authentic ER.	71
4.6	Filtered metrics of nodes in suspicious list.	74

4.7	Detection delay of ER manipulation.	77
4.8	SDBG's detection accuracy under varying attack settings.	78
4.9	SDBG's false positive rate under varying attack settings.	79
4.10	SDBG's detection time under varying attack settings.	80
4.11	Detection accuracy and false positive in Prophet under varying thresholds.	81
4.12	Detection time of Prophet under varying threshold setting.	82
4.13	SDBG vs other schemes for individual attack.	84
4.14	SDBG vs other schemes for collusion attack.	86
4.15	SDBG's data overhead size.	88
4.16	SDBG detection performance under varying window size.	89
4.17	SDBG under varying attack period T	91
4.18	SDBG's detection accuracy in behavior-switching attack.	92
5.1	Factors of flooding attack.	97
5.2	Identity-based cryptography signatures and batch verification	100
5.3	ER manipulation strategies.	104
5.4	Detecting ER manipulation.	106
5.5	Manipulating message count.	107
5.6	Manipulating replica count.	109
5.7	Detecting inconsistent assignment of replica count.	111
5.8	Attacker's trade-off between the number of flooded replicas and the rate of reusing count.	112
5.9	ER manipulation detection time and accuracy.	119
5.10	Performance of detecting message count reuse.	120
5.11	Performance of detecting burst-limit violation.	121
5.12	Forwarding priority of normal and malicious nodes over time.	123
5.13	Delivery performance of normal nodes with and without fairness policy (FP).	123
5.14	Performance of detecting replica count reuse.	124
5.15	FDER's detection of message count reuse attack when varying window size.	125
5.16	FDER's detection of replica count reuse attack when varying window size.	125
5.17	FDER's ER size.	126

6.1	Recognition protocols executed by node B	133
6.2	Areas associated with local danger.	147
6.3	Aggregating transitive trust.	153
6.4	Decision tree of linkability-based entity trust.	154
6.5	Relationship of trust and privacy	156
6.6	Correct decision rate.	157
6.7	Detection rate of malicious nodes.	158
6.8	Correct decision rate under varying probability of enabling id linkability.	158
6.9	Correct decision rate under varying probability of enabling trust linkability.	159

List of Tables

3.1	Performance of SNDW vs prohibited-topology method	51
4.1	Notation of SDBG's parameters.	66
4.2	SDBG's detection rate of ER manipulation.	77
6.1	Bloom filter based PSI.	136
6.2	PSI-CA: Parameters and functions	137
6.3	PSI-CA: Procedure (all computations are mod p).	138
6.4	Identity linkability establishment protocol.	139
6.5	Identity recognition protocol.	141
6.6	Trust recognition protocol.	143

List of Abbreviations

ABBREVIATIONS**FULL EXPRESSIONS**

ADT	All Distance Test
ALRS	Adaptive Linkability and Recognition Scheme
ATMS	Adaptive Trust Management Scheme
AODV	Ad hoc On-demand Distance Vector routing
BF	Bloom Filter
CA	Certificate Authority
DKM	Distributed Key Management
DoS	Denial of Service
DSDV	Destination-Sequenced Distance-Vector routing
DSR	Dynamic Source Routing
DSRC	Dedicated Short-Range Communications
DTN	Delay Tolerant Network
EMR	ER Manipulation Ratio
ER	Encounter Record
FBIDM	Ferry-Based Intrusion Detection and Mitigation
FDER	Flooding Detection based on Encounter Record
FP	Fairness Policy
FR	Forwarding Ratio
GERDB	Global ER Database
GM	Group Manager
GPS	Global Positioning System
GSIS	Group Signature and Identity-based Signature
IBC	Identity-Based Cryptography
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
KL	Kullback-Liebler
KGC	Key Generator Center
LTE	Long Term Evolution

MAC	Media Access Control
MAD	Mutual Authentication with Distance-bounding
MANET	Mobile Ad hoc NETwork
MDS	Misbehavior Detection Scheme
ML	Meeting List
MM	Membership Manager
MR	Message Record
MS	Meeting Summary
MUTON	Mutual Correlation Detection Scheme
NS	Network Simulator
NTT	Neighbor Number Test
OBU	On Board Unit
ONE	Opportunistic Network Environment simulator
PKI	Public Key Infrastructure
PMDS	Probabilistic Misbehavior Detection Scheme
PP	Pseudonym Provider
PRoPHET	Probabilistic Routing Protocol using History of Encounters and Transitivity
PSI	Private Set Intersection
PSI-CA	Private Set Intersection CArdinality
PSN	Pocket Switched Network
RR	Relaying Ratio
RREQ	Route REQuest packet
RSU	RoadSide Unit
RTT	Round Trip Time
SBAP	Secure Broadcast Authentication Protocol
SECTOR	SECure Tracking Of node encounteRs
SFR	Self-Forwarding Ratio
SNDW	Statistical Neighbor-count-based Detection of Wormhole attack
SDBG	Statistical Detection of Blackhole and Greyhole attack
TA	Trusted Authority
TCA	Trust Checking Agent
VANET	Vehicular Ad hoc NETwork
WiFi	Wireless Fidelity
WLAN	Wireless Local Area Network
WSN	Wireless Sensor Network

Abstract

Novel network paradigms such as delay tolerant network (DTN), vehicular ad hoc network (VANET) have recently emerged to accommodate new wireless network application scenarios. DTN, designed to support networks characterized by intermittent connectivity and long delay, has been widely embraced for ad hoc applications in extreme environments such as battlefield, disaster area, rural area, etc. VANET allows vehicles travelling on the roads to exchange data that can improve traffic safety and convenience. Despite the advances in operational standardizations of DTN and VANET, security and privacy remain as major problems to their deployments in practice. This motivates the author to study and resolve the unaddressed security and privacy issues in DTN and VANET in this dissertation.

First, the author proposes SNDW to detect and localize wormhole attack in DTN. Existing works rely on high connectivity or end-to-end path information which are not available in DTN. In SNDW, infrastructure nodes are deployed to monitor the number of neighbors in the areas assigned to them. Since the attacker forwards beacons from one location to another, the number of neighbors around the wormhole endpoint increases. An infrastructure node within the wormhole's transmission range can detect a significant increase of neighbor counts, despite the low density and connectivity in DTN. Upon detection, it can alert the neighborhood about the attack and further pinpoint the wormhole locations.

Second, the author proposes SDBG to detect collusion blackhole and greyhole at-

tacks in DTN. Existing works could handle the attacks launched by individuals but fail to detect adversaries that cooperate to cheat the defense system. In SDBG, encounter record (ER) is created by a node to record the list of messages it sends and receives in an encounter with another node. Upon encounter, nodes exchange their recent ERs so that they can evaluate one another's behavior. Forwarding ratio metrics are defined to capture the attacker's features such as receiving messages more than forwarding and forwarding messages mostly generated by themselves. A node calculates the forwarding ratio metrics of the peer from its ERs to detect the individual attacker. However, adversaries might avoid being detected by colluding with one another to forge ERs that promote their forwarding ratio metrics. Observing that fake ERs are created with higher frequency and higher numbers of sent messages than normal ERs, the author designs a robust algorithm that filters fake ERs and detects colluding attackers.

Third, the author proposes FDER to detect flooding attack in DTN without imposing strict rate-limit policy like previous works. Instead, burst-limit policy, which allows nodes to generate burst traffic in a short period, is introduced to accommodate the nodes' on-demand transmissions. Each node also uses ERs to infer a peer's new message transmission rate over time and the number of forwarded replicas per message, thereby detecting if the peer exceeds the thresholds of messages or replicas. Burst-limit policy provides flexibility to nodes with high-demand traffic but may affect the delivery performance of other nodes with lower-demand traffic. To ensure the performance fairness, the author designs a forwarding policy, in which nodes sending out new messages at high rate will have lower forwarding priority in the network.

Finally, the author attempts to address the trade-off between trust management and privacy in VANET. Existing works only focus on trust or privacy separately or have not provided a satisfactory solution to both issues. The author proposes a secure and flexible framework, including two components, ALRS and ATMS, to enable vehicles

to manage privacy and trust respectively. ALRS supports vehicles to recognize either identities or trust levels of other vehicles despite them updating the pseudonyms to protect their privacy. The linkability information is kept confidential from external attackers and unauthorized internal nodes using encryption and private set intersection technique. Besides, the linkability can be revoked easily to prevent nodes from being traced by other internal nodes. ATMS is a context-aware trust management scheme for nodes to make more informative evaluation of the received data, using the entity trust enabled by ALRS.

Chapter 1

Introduction

1.1 Background and Research Methodology

The rapid development of wireless technologies and portable devices (such as sensors, laptops, mobile phones, etc) has given rise to a new type of network called Mobile Ad hoc Networks (MANET) [22]. MANET consists of mobile nodes, which are free to move around and communicate with one another using wireless links, without reliance on centralized infrastructure. The concept of MANET has been integrated with emerging network paradigms such as delay tolerant network, vehicular network, to accommodate new application scenarios.

Delay tolerant network (DTN) [1–3] is developed to cope with networks characterized by link disruption, partitioned connectivity, high error-rate, and asymmetric data rate. A few examples of such challenging networks are the communication in extreme environments like outer space, battle fields, disaster areas, highway roads, etc. Thus DTN has been widely embraced for many practical applications such as sensor networks for wildlife tracking [4], Pocket Switched Network (PSN) [5], very high-speed

vehicular network [6] and low-cost Internet connection in remote villages [7].

Vehicular ad hoc network (VANET) [12, 13] is a special type of MANET formed by vehicles travelling on the roads. VANET has attracted a great deal of research as the potential solution to safer, more efficient and more convenient traffic. The communication among vehicles enables various vehicular applications for road safety [14–17], driver assistance and infotainment [18–21]. For example, in cooperative collision warning application [14], vehicles that are aware of road accidents can disseminate the warnings to other vehicles in the vicinity so that they can take actions in advance and avoid the danger.

Similar to MANET, DTN and VANET are also vulnerable to various security and privacy issues for several reasons. First, the broadcast nature of radio transmission permits the adversaries to easily overhear messages which are not intended for them. By eavesdropping, adversaries can infer the nodes' locations for unauthorized tracking, violating their privacy. Second, the lack of infrastructure complicates the security issue since there are no centralized facilities to monitor the network traffic to detect and prevent nodes' misbehaviors. Moreover, mobile nodes often have limited resources of energy and computing, making them vulnerable to exhaust their batteries in the presence of attacks. Without proper security mechanisms, MANET as well as DTN and VANET cannot function as desired.

Due to the distinguished features and applications, DTN and VANET bring about new challenges in the security and privacy issues. Like MANET, DTN also suffers from various types of attacks [23] such as wormhole, blackhole, greyhole, flooding, spoofing. However, the lack of continuous path and limited connectivity resources in DTN make the detection of these attacks harder than that in a well-connected general MANET. Since VANET involves real-life vehicles, the privacy issue in VANET becomes more critical and sensitive than MANET. If the location privacy of vehicles are not well-

protected, the drivers' identities and profiles may get revealed when they use VANET applications. This discourages them from joining VANET, making it hard to deploy VANET in reality.

The research methodology adopted is as follows. The author first conducts comprehensive literature reviews about the security and privacy issues in DTN or VANET and identifies the particular unsolved challenges. Next, the author proposes the corresponding schemes to the outlined problems. The protocols' designs are described in details with highlight on the novelty of the proposal. Finally, the author implements extensive simulations to evaluate the performance of the proposed schemes in comparison with previous works.

The simulator tools used are Network Simulator 3 (NS3) [27] and Opportunistic NEtwork simulator (ONE) [28, 29]. NS3 is a sophisticated discrete-event-driven network simulator written in C++, which enables simulating wireless/IP networks including WiFi, WiMax, LTE. ONE is a simple simulation environment written in Java, which focuses on modelling opportunistic networks by providing map-based node movement models and various routing algorithms specific to DTN. Both NS3 and ONE are well-documented open source projects, which provide a free but powerful platform to evaluate the performance of the proposals in a controllable and low-cost manner.

1.2 Motivation and Scope

1.2.1 Wormhole attack in DTN

Wormhole attack was first realized as the threat to MANET in [30]. In this attack, one or more malicious nodes collaborate with one another to record messages at one location and then replay them at another location. As depicted in Fig. 1.1, two wormhole

Figure 1.1: Wormhole attack.

endpoints M_1 and M_2 , connected by a dedicated link, can tunnel the messages from one node to the other. Attackers can exploit wormhole in many ways. For example in MANET, M_1 and M_2 can disrupt the routing discovery phase of routing protocols such as AODV [24] and DSR [25] as follows. By using the tunnel, M_1 and M_2 can advertise the route $S - A - M_1 - M_2 - B - D$ between the source node S and destination node D which is shorter than the valid route $S - A - H - I - J - K - B - D$. Thereby, malicious nodes can attract data to traverse the wormhole and then drop them, leading to decreased message delivery.

Another exploitation approach is to disrupt the neighbor discovery process by tunnelling beacon messages of two distant nodes A and B . In that way, A and B can receive beacon messages of each other and believe that they are direct neighbors. In MANET using DSDV [26], in which routing tables are updated based on neighbors information, A cannot forward data to B and vice versa because they are actually not in the transmission range of each other, leading to routing failures.

In DTN, fake neighbor information can seriously affect the routing protocols that are based on encounter history to evaluate the routing capability of nodes. For example

in DTN using Prophet [9], A 's delivery probability to B will be amplified upon the fake encounter between them due to the wormhole, even though A may actually have very low possibility of meeting and forwarding data to B . Moreover, due to the transitive property of Prophet, the nodes that frequently encounter A will also have their delivery probabilities to B increased. This will distort the routing metrics of multiple nodes and make nodes forward data to the incorrect candidates, leading to wasted resources and reduced delivery performance. Due to the scarce resources, DTN is more vulnerable to wormhole attack than MANET. At the same time, it is more challenging to detect wormhole in DTN.

Previous works attempt to detect wormhole attack in MANET by exploiting the mismatching properties or changes that appear due to wormhole such as location [30–32], time [30,33–35], statistics of network parameters [36–39] or graph structure [40,41]. However, the methods based on location and time require additional hardware per node or tight clock synchronization which are expensive for distributed network of DTN. Meanwhile, the graph and statistical-based methods either rely on end-to-end routing path information which is hard to obtain in DTN or only work well in high-density network which is not the case of DTN.

To address the wormhole attack in DTN, the author first proposes the statistical parameter that changes in the appearance of wormhole and is independent of the network density. The author then exploits it to design a scheme, named SNDW (Statistical Neighbor-count-based Detection of Wormhole attack), that can detect and localize the wormhole attackers.

1.2.2 Blackhole and greyhole attacks in DTN

In blackhole attack, the adversaries drop all the received messages even if they have enough buffer storage. Meanwhile in greyhole attack, the adversaries only drop a fraction of received messages to avoid arousing suspicion and detection from other normal nodes. The dropping misbehavior will decrease the overall message delivery and waste the resources of intermediate nodes that have carried and forwarded the dropped messages. The lack of continuous path and limited connectivity resources in DTN make the detection of these attacks more challenging than that in a well-connected MANET.

Blackhole and greyhole attacks in DTN have been studied in [47–53]. [47, 48] require trusted ferries to check if nodes arbitrarily increase their delivery probabilities to absorb more data, which is preliminary to the dropping attack. [49] designs a trust-based framework in which the message forwarding behavior is acknowledged and rewarded with reputation. [50] depends on a trusted authority to investigate nodes based on the evidences of their forwarding tasks, contact opportunities and actual forwarding behaviors. [51–53] make use of nodes' encounter records to monitor nodes' behaviors and detect the attack.

While the schemes mentioned above can detect individual attackers well, they cannot handle the case where attackers cooperate to avoid the detection. For example, in an individual detection scheme, nodes are evaluated by their histories of encounters with other nodes, i.e. encounter records (ERs). A node is judged as malicious if its forwarding ratio, which is the ratio between the total number of sent and received messages, is lower than a threshold. This works since attackers tend to drop received messages instead of sending them out. However, if attackers cooperate to boost the forwarding ratio by creating forged encounter records for one another, they can avoid

being detected by the scheme. This misbehavior is referred to as collusion attack. As shown in Fig. 1.2(a), n_3 can detect m as malicious when judging its authentic records with n_1, n_2 . However, Fig. 1.2(b) shows that by colluding with m' , m can produce fake records to trick n_3 to judge it as normal.

The author observes that colluders (m and m') have to create a large number of forged records to persistently avoid the detection. Besides, these fake records have to include high number of sent messages to compensate for the dropped messages. These patterns enable the detection of the collusion attack as in Fig. 1.2(c). When m encounters n_3 , it submits the forged encounter records window which includes contacts with n_1, m', n_2 and m' . The high encounter frequency and number of sent messages between m and m' triggers n_3 to suspect m' as colluder of m . To confirm that, n_3 excludes the suspicious fake records between m and m' from the submitted history of m . In this way, it will obtain m 's actual encounter history which includes contacts with n_1, n_2 only. The real forwarding ratios inferred from the authentic window allow n_3 to detect m as malicious. Exploiting all the observations mentioned above, the author proposes SDBG (Statistical Detection of Blackhole and Greyhole attack) – a robust detection scheme that addresses both individual and collusion blackhole and greyhole attacks.

1.2.3 Flooding attack in DTN

In DTN, messages are routed to the destination in a “store-and-forward” fashion: each message is first stored in the permanent storage of the source node or relay nodes and then forwarded opportunistically to the optimal next hops or the destination node. To increase the delivery rate, most routing protocols allow data to be replicated and propagated in different paths to the destination. However, malicious nodes may utilize

Figure 1.2: Examples of attacks and detection schemes. n_1, n_2, n_3 are normal nodes. m and m' are malicious nodes.

this feature to launch flooding attack. By sending out large number of new messages, the adversaries deplete the scarce resources of legitimate data, thus severely degrading the delivery performances of benign nodes.

While flooding attack is well addressed in wired networks [54–56] and MANET [59–62], these solutions cannot be directly applied to protect mobile DTN which is distributed and lacks connectivity. Many solutions are proposed to prevent flooding attack in DTN. Ansa et. al. [63] designs a lightweight authentication scheme. Lee et. al. [64] regulates the dropping policy for a node’s message queue. However, [63,64] cannot detect internal flooding attackers. Other works like [65,66] limit nodes’ message generation rates and detect internal adversaries that exceed the rate limit. However, the rate-limit policy is not flexible enough for networks with unpredictable transmission demand.

The author introduces burst-limit policy to restrict flooding attack while flexibly accommodating the nodes’ transmission demands. The author proposes FDER (Flooding Detection based on Encounter Record) – a scheme based on encounter records to detect the flooding attackers that violate the burst-limit policy. Finally, the author designs a forwarding policy to ensure the performance fairness among nodes with different transmission demands.

1.2.4 Trust and privacy trade-off in VANET

VANET can only improve the traffic safety if the propagated warnings are assured to be trustworthy. For example, a malicious vehicle may report a non-existing collision to make the vehicles behind react by braking abruptly, which could possibly cause a chain collision among these vehicles. To deal with such fraudulent messages, vehicles need to evaluate the trustworthiness of the entities that send/relay data and the credibility

of the data before deciding to believe in them.

While vehicles may enjoy the benefits of VANET applications, they also expose themselves to the threat of location privacy when they cooperate to propagate the messages. Attackers can eavesdrop their messages and infer their location histories for unauthorized tracking. Without privacy-protection schemes, vehicles may be deterred from joining VANET, making it hard to deploy VANET in reality.

Researchers have proposed a number of trust management schemes in VANET. Vehicles evaluate the credibility of the received data by objectively verifying the data against practical and intuitive models [70–75] or fusing the peers’ opinion towards the data in consideration of the peers’ reputations [76–80]. Existing studies on privacy in VANET suggest vehicles use pseudonyms [81, 82, 84, 85] or group signatures [86–88] that are uncorrelated to their real identities to enable them to stay anonymous and authenticated. Unfortunately, these works only handle either trust or privacy without considering both issues jointly.

Though trust and privacy are both important, there is a tradeoff between them. When vehicles change pseudonyms over time to protect privacy, they cannot recognize one another, making it hard to build peer trust. When a node changes pseudonym frequently, other nodes will have insufficient observations of its behaviors to evaluate the entity trust accurately. However, low frequency of pseudonym change allows external eavesdroppers to track vehicles more easily. To enable both trust management and privacy enhancement, [96] proposes a protocol that allows a node to temporarily track a neighbor to update the trust despite the use of pseudonyms. However, the proposal requires vehicles to reveal their identities in plain text to request for tracking. The leakage of this private information enables adversaries to trace the nodes’ locations and threaten their privacy.

Aiming at mitigating the trade-off between trust and privacy in VANET, the author proposes a framework for vehicles to manage and balance privacy and trust in a flexible manner. The framework includes two components: Adaptive Linkability and Recognition Scheme (ALRS) and Adaptive Trust Management Scheme (ATMS). ALRS allows vehicles to grant or revoke selective nodes with the ability to recognize them by identity or trust level. By managing the recognition grant, vehicles can protect their privacy adaptively according to the context, i.e. either hiding identity or trust level when privacy is prioritized or revealing identity when trust management is required. ATMS utilizes the granted recognition information to allow vehicles to verify the data sent by other nodes and update their reputations.

1.3 Major Contributions

The authors have made the following contributions to address the security and privacy issues identified in Section 1.2.

1.3.1 Detecting wormhole attack in DTN

The author first studies if the neighbor count parameter, which is the number of neighbors discovered by nodes in the network, could play as indicator of wormhole attack in DTN. Since wormhole attacker forwards beacons from one location to another, the number of neighbors around the wormhole endpoint increases. Despite the low absolute neighbor counts due to the sparse density in DTN, the neighbor counts change around the wormhole is distinguishable from that in other areas. Thus, the neighbor counts in a small area could be used to detect wormhole attack in DTN.

The author next proposes SNDW that relies on infrastructure nodes to monitor

the neighbor counts in their assigned areas. An infrastructure deployment layout and monitor schedule are designed to ensure they can cover the network area. If an infrastructure node experiences an abnormal increase of neighbor count compared to the normal count, it alerts the network about the attack. The alerting node can cooperate with other infrastructure nodes to further reduce the possible locations of wormhole within its monitoring area.

The author implements simulation to evaluate the performance of SNDW. The results show that SNDW detects wormhole in DTN with higher accuracy and lower false positive compared to the related work based on prohibited topology, especially in high-speed network such as vehicular network.

1.3.2 Detecting blackhole and greyhole attacks in DTN

The author first designs a procedure to detect individual blackhole and greyhole attacks. When two nodes encounter, they create an encounter record (ER) to record their encounter and the lists of messages sent and received. Encountering nodes also exchange their most recent ERs and use them to judge one another's behavior. The judging node computes the forwarding metrics of the judged node and checks if they violate the predefined thresholds. If the violation is detected, the judging node decreases the trust towards the judged one.

Next, the author designs an algorithm to detect colluding attackers, which involves two phases: suspicion and confirmation. The first phase lists the suspicious colluders with the judged node. Nodes are suspected if the total number of messages they receive from the judged node is higher than a threshold. However, normal nodes, who have close relationship with the judged node, may also encounter and exchange messages a lot with it. To avoid the misjudgement, the second phase is introduced to verify the

suspicion. The judging node omits the ERs between suspicious colluders and only uses the remaining ERs to assess the forwarding ratios (as in individual detection).

Simulation is implemented to evaluate SDBG's performance. It shows that SDBG can detect collusion attack with higher accuracy and lower false positive rate compared to state-of-the-art schemes.

1.3.3 Detecting flooding attack in DTN

The author introduces burst-limit policy to replace the rate-limit policy that prevent nodes from flexibly sending messages on demand. Each node's normal traffic pattern is still characterized by rate-limit but the node is allowed to have a small and short burst of new message which exceeds the rate-limit. However, if the node generates a large burst continuously for a long period, it is deemed as a flooding attacker and will be detected by FDER.

The author designs FDER to detect attackers that flood messages and replicas. To monitor the number of messages and replicas that each node has sent out in DTN in distributed manner, FDER lets each node record what it sends during an encounter with another node in the ER created between them. Similar to SDBG, encountering nodes exchange their most recent ERs so that they can judge one another's behavior. The judging node computes the number of messages and replicas sent by the judged node during its past encounters and checks if they violate the thresholds of messages and replicas.

Enabling burst transmission raises the concern that the nodes with high transmission demands could occupy more network resources than those with low demands. To ensure fairness among nodes with different traffic patterns and to mitigate the effect of the flooding misbehavior before it is detected, the author designs a forwarding policy

to lower the forwarding priority of messages which is generated by nodes with high generation rate. This additional feature can be easily incorporated into existing DTN routing protocols.

The author implements simulation to evaluate FDER's performance. It shows that FDER outperforms the state-of-the-art scheme in detecting the burst-limit violation and the forwarding policy ensures the delivery performance fairness among nodes.

1.3.4 Balancing trust management and privacy in VANET

The author designs ALRS to provide flexible recognition in a secure and private manner. Having vehicles exchange their identities when they encounter is a simple and straightforward solution to let them recognize one another. However, this process cannot support the selective recognition and the recognition by trust level. To resolve this issue, the author suggests that a pair of vehicles confidentially exchange their identities and agree on a shared link value when they first meet. Thereafter, they rely on link values to recognize each other and to grant/revoke the recognizability to selected nodes. Specifically, a vehicle can recognize a peer's identity by determining the intersection between their lists of link values. It can recognize if a peer is trusted by determining if there is any common item between the peer's list of link values and its list of values linked to the trustworthy nodes. The vehicle can easily switch between allowing and disallowing a peer to recognize it by including or excluding the shared link value from its list. To protect the private lists of link values in intersection operations, Private Set Intersection (PSI) and Private Set Intersection Cardinality (PSI-CA) protocols are used.

The author also designs ATMS to support adaptive trust management under the privacy context set by ALRS. A vehicle adapts the approach to evaluate event data

based on its distance to the data originator. Specifically, if the distance is near enough, it can verify the event by directly observing if the movement response of the data originator is consistent with the event. Otherwise, it can collect different event reports and aggregate them in consideration of its trusts towards the data originators, which may not be obtainable directly under the privacy context. Thus the vehicle needs to adapt the entity trust setting based on the linkability level granted by other nodes in ALRS. To achieve this, the author designs a protocol for vehicle to derive the entity trust value from the available linkability information.

The author simulates VANET scenario to evaluate ATMS's performance under different settings of recognition enabled in ALRS. The simulation results show that the framework achieves high rate of accurate node's decision towards the reported event data and high detection rate of malicious nodes which deliberately report the wrong events.

1.4 Thesis Organization

The rest of the thesis is organized as follows:

- **Chapter 2** reviews the previous works on the related topics. Various security issues including wormhole, blackhole, greyhole, flooding attacks are discussed together with existing proposals to defend against these attacks in DTN. Trust management and privacy protection schemes for VANET are also presented.
- **Chapter 3** introduces SNDW that is capable of detecting and localizing wormhole in DTN. The mechanism of SNWD is described and the performance is analyzed and evaluated using network simulations.
- **Chapter 4** presents SDBG which aims at detecting colluding blackhole and

greyhole attack in DTN. How SDBG works and performs are shown.

- **Chapter 5** describes FDER to detect flooding attack in DTN. The protocol details and performance results using simulation are included.
- **Chapter 6** sheds light on the trade-off between trust management and privacy issues in VANET. An adaptive trust and privacy management framework is proposed to resolve the trade-off. The protocol design and performance evaluation are elaborated.
- **Chapter 7** concludes the thesis and sketches out future research directions.

Chapter 2

Literature Review

This chapter surveys the existing literature for each of the problems outlined in Section 1.2. Current approaches, their advantages and limitations are discussed. How the author's works are distinguished from the previous efforts are also highlighted.

2.1 Wormhole Attack

2.1.1 Detecting wormhole attack in MANET

Existing solutions can be classified into four types:

1. Location-based methods
2. Time-based methods
3. Statistical-based methods
4. Graph-based methods

Location-based methods

In [30], before sending a packet, the sender attaches with the packet the leash information that includes its current position and the sending time so that the receiver can estimate the maximum distance between them. If the estimated distance is too large, the packet is detected as being forwarded through wormhole endpoints. This method requires location service such as GPS, loosely-synchronized clocks and some authentication scheme to protect the leash information.

Directional antennas are used to detect false neighbor relationship due to wormhole attack [31]. It is assumed that if one node sends a packet in a given direction, then its neighbor will get that packet in the opposite direction. Therefore, two nodes are authentic neighbors if they can verify the valid directions of sending and receiving using the antennas. This solution suffers from directional errors and entails all nodes to be equipped with additional special hardware.

Beacon nodes, which already know their absolute locations via GPS or manual configuration, are utilized to detect wormhole attack [32]. At the beginning, each beacon node broadcasts a probe message to the network. Receiving this message, mobile node increments the hop count in the content before rebroadcasting it. When the probe message reaches another beacon node, the receiver checks if the hop count is consistent with the actual distance between it and the sending node. This scheme is simple and fast but the broadcast overhead over multiple hops can be large. Besides, it requires very dense layout of beacon nodes (0.008 per m^2).

Time-based methods

Temporal leash [30] constrains the packet life-time to prevent packets from travelling over a large distance due to wormhole. When sending a packet, the sender inserts

the expiration time of the packet based on the sending time so that the receiver can tell if the packet has travelled too far based on the transmission time and speed. The disadvantage of this scheme is the requirement of tightly synchronized clocks for all nodes which is expensive to achieve.

SECTOR [33] depends on Mutual Authentication with Distance-bounding (MAD) protocol, in which the first party sends a bit to challenge the other party to response with a bit immediately. Based on the difference between the sending and receiving time, the first party computes the upper bound of the distance to the other party. In SECTOR, the second party can start sending its own challenge once it receives some bits from the first party so that both sides can estimate their mutual distance simultaneously. This scheme requires nodes to have highly accurate and sophisticated clocks to measure the time to nanoseconds accuracy.

WORMEROS [34] is a two-phase framework to detect wormhole attacks in MANET. The first phase discovers suspicious evidence of wormhole attack using inexpensive technique of round-trip time (RTT). A neighbor whose RTTs is abnormally higher than the average RTT is suspect. It will get challenged in the second phase using expensive and accurate technique such as frequency hopping or TrueLink [35] to confirm the presence of wormhole. The two-phase scheme reduces the cost of using expensive technique all the time. However, the RTT-based assumption is not true for out-band wormhole where wormhole endpoints use super fast link to tunnel the message.

Statistical-based methods

[36] detects wormhole attack in WSN using Chi-square test on parameters that are possibly changed due to the presence of wormhole. Two mechanisms are proposed: Neighbour Number Test (NNT) to detect the increase in the number of neighbours

Figure 2.1: Hypothetical VS real distribution of network parameters.

of the sensors and All Distances Test (ADT) to detect the decrease of the lengths of the shortest paths between all pairs of sensors. Fig. 2.1 shows an example of the difference between hypothetical and real distribution (in the presence of wormhole) of those parameters. The scheme requires each node to send neighbourhood information to a server to implement the detection algorithm. The scheme cannot localize the wormhole. Besides, using such server causes large amount of communication overhead.

[37] proposes another statistical approach in which each sensor collects the recent history of the number of neighbours and detects if the current neighbour counts exhibit abnormal increases compared to the normal ones using the Kullback-Liebler (KL) divergence. Fig. 2.2 shows an example of change in statistical data over time, indicating special case of true positive and false positive wormhole detection. This solution is distributed and online so no overhead is incurred. However the algorithm performance in the case of non-uniformly-distributed network and sparse density is in question.

SAM [38] and SA-TC [39] are other schemes using statistical analysis to detect the network change when wormhole appears. Both schemes exploit the fact that if the

Figure 2.2: Example of statistical change in WSN.

network is attacked, the link tunnelled through the wormhole endpoints will potentially appear with higher frequency in the multi-path routing. These methods are more suitable for networks that use multi-path routes and have available information of end-to-end path.

Graph-based methods

[40] models the network as a connectivity graph with edges and vertices and exploits the edge-clustering coefficient to detect wormhole. For each edge, this coefficient is defined as the ratio of the number of cyclic structures of order g built on that edge and the maximal possible number of such structures. If an edge involves a wormhole endpoint, that edge could not have sufficient number of cyclic structures and the wormhole is thus detected. The scheme is able to detect and isolate individual wormhole attacker but cannot deal with multiple cooperating wormholes. Besides, it incurs high false positive rate in low-density network.

[41] depends on the packing argument which states that a fixed region cannot pack

too many independent nodes, i.e. any pair of nodes whose distance is larger than a certain value. This means two non-neighbour nodes can only have at most f_k common independent k -hop neighbours. Otherwise, the packing argument is violated, indicating the appearance of wormhole. To detect wormhole, the scheme requires nodes to discover the k -hop neighbours. For dense network, it can work well with $k = 1$. However, for sparse network, it requires collecting multiple-hop neighbour information and incurs high overhead for nodes.

2.1.2 Detecting wormhole in DTN

The works described above are originally targeted at well-connected wireless networks like MANET and WSN. They cannot be applied directly to DTN because they either require network resources that are limited in DTN nodes or rely on connectivity information or end-to-end routing path information which is hard to obtain in DTN.

Prohibited-topology method [42] is a graph-based method to detect wormhole in DTN which depends on the forbidden topology seen in Fig. 2.3. A and B are two wormhole endpoints and C, D, E are normal nodes. When the transmission range is R , node C has two independent neighbors D and E which means the distance between D and E is larger than R . When node C reduces its transmission range to $R/2$, if nodes D and E are still independent neighbors of C , this violates the triangle inequality as follows: $CD + CE < R/2 + R/2 < R < DE$. This violation is prohibited in legitimate network and implies the presence of wormhole attack.

The scheme is advantageous in that it is simple, distributed, incurs low overhead and involves few nodes (three nodes in this case) in the prohibited topology. Thus, it is suitable for DTN with low density and resource-constrained nodes. However, for imperfect network model with propagation loss or network with high-speed nodes, the

Figure 2.3: Prohibited topology due to wormhole.

connectivity information may not be maintained exactly, leading to high false positive rate.

Unlike previous works, SNDW does not require extra hardware per node and can work despite the low density of DTN. To detect wormhole, SNDW exploits the statistical metric of neighbor counts which is independent on the nodes' speed. As a result, the false positive rate of SNDW is much lower than the prohibited-topology.

2.2 Blackhole and Greyhole Attack

Cryptography provides networks with basic security services such as authentication, message integrity and non-repudiation. Several works [43–46] have focused on designing cryptography schemes suitable in DTN and applying cryptography to the links to secure against malicious nodes. The cryptography approach can protect networks from unauthenticated external adversaries. However, it is not enough to defend against authenticated internal adversaries that launch such attacks as blackhole and greyhole

attacks. Therefore, researchers have proposed misbehavior detection schemes to detect and mitigate internal attackers. Existing schemes can be classified into two types:

1. Methods based on delivery predictability records
2. Methods based on forwarding receipts and encounter records

2.2.1 Methods based on delivery predictability records

The routing protocol Prophet [9] uses the delivery predictability as the routing metric to measure the capability of one node to forward a message towards the destination. Utilizing the unsecured Prophet, an adversary can arbitrarily increase the delivery predictabilities from itself to the targeted nodes. Thus the adversary will increase the chance to be selected as the next hop and thus can absorb more packets destined to the victims to launch the blackhole attack.

To detect the above misbehaviour, FBIDM [47] utilizes ferries which are special authorized mobile nodes that follow a fixed route to collect the delivery predictability information from other nodes. Each node i is required to keep the history of the last M values of its delivery predictability P_{ij} to other node j and share these records with the ferry when they meet each other. The ferry checks the consistency of the delivery predictability of two nodes P_{ij} and P_{ji} using their respective history data. If P_{ji} is much larger than P_{ij} , it is highly possible that node j is an attacker and thus should be isolated.

MUTON [48] is an enhancement method to FBIDM by exploiting two factors. First, MUTON considers the transitivity of Prophet to estimate the changes in delivery predictability. Second, MUTON calculates the estimated delivery predictability of one node by correlating the information collected from other nodes. The estimation is then

compared to the predictability claimed by that node. As a result of these improvements, MUTON can detect the attack faster and at lower false positive than FBIDM. However, both FBIDM and MUTON require trusted monitoring authorities and only work for DTN using Prophet.

2.2.2 Methods based on forwarding receipts and encounter records

Forwarding receipt is the secured evidence to acknowledge that a node has forwarded a message to other relays. [49] suggests a distributed trust-based framework based on forwarding receipts and reputation system as illustrated in Fig. 2.4. First, node A forwards a message m to B , i.e. B is assigned with the forwarding task of m . Next, if node B transfers m to C , C generates the forwarding receipt to acknowledge the forwarding behavior of B . Finally, when node C meets A , it sends A the forwarding receipt so that A increases the reputation to B . Since blackhole attacker drops the message instead of forwarding, it has no forwarding receipts, leading to its low reputation and the adversary becomes untrusted. However, the scheme cannot detect an attack in which nodes B and C cooperate to create fake forwarding receipts.

Figure 2.4: Blackhole defense using reputation and forwarding receipt.

Routing algorithms in DTN such as Prophet, Maxprop utilize the past contacts of

a node to infer its routing capability. [51] proposes encounter ticket to prove that two nodes has met. Since encounter ticket is secured by the signatures of both sides, attackers cannot claim non-existent encounters and abuse them to forge routing metrics to attract the data to launch the blackhole attack. However, attackers can tailgate the targeted victim for a period to boost the routing metric. [51] thwarts the tailgating attack by inferring the actual routing competency of nodes from their histories of encounter tickets using Dempster-Shafer theory. The method can only prevent the adversaries from falsifying the routing metrics but cannot detect the dropping misbehavior, making it insufficient to thwart the blackhole attack.

PMDS [50] proposes that a Trusted Authority (TA) investigates if a node commits routing misbehaviors such as blackhole/greyhole attack. There are three types of evidences necessary for judging a node M : (1) delegation history D which records all the messages forwarded to M as its routing tasks, (2) forwarding receipt history F which records all the messages forwarded by M to prove that M has forwarded them, (3) contact history E which records the encounter between M and other nodes. Nodes create and secure these evidences with cryptography and submit them to TA upon the start of an investigation. TA integrates the evidences D , E and routing protocol R to produce the ideal forwarding candidates W for node M . If the forwarding history F of node M does not match W , M is suspected as an adversary. PMDS requires centralized server which might become the central point of attack and incurs high overhead due to the investigation.

MDS [52] combines forwarding receipt and encounter ticket into the same record called encounter record. Encounter record contains not only the meeting time and nodes' signatures but also the list of exchanged messages during the encounter. When two nodes meet, they judge each other based on the peer's encounter record history which reflects the overall message forwarding percentage θ and the own message for-

warding percentage ϕ of the peer. As the malicious nodes often drop packets of others but prefer transmitting messages generated by themselves, the adversaries tend to have a low value of θ and a high value of ϕ . Thus these metrics can play as indicators of the misbehaviour. The suspected attacker will have its trust reputation decreased by the judging node and when the reputation falls below a certain threshold, it is blacklisted locally. However, MDS cannot detect the adversaries that cooperate to create fake encounter records to boost the defined metrics.

In [53], an encounter record includes the initial message buffer, sent and received messages during an encounter between two involved nodes. A misbehaving node can drop packets received from the previous contacted node, which will be detected by the next contacted node as it can observe the disappearance of the packets in the current buffer record. To hide the dropping misbehavior, the adversaries can cooperate to create fake encounter records as evidence that they have forwarded the message. Due to the fake records, one of the collusion members should keep the message which has actually been dropped. Thus this member will get ultimately detected and cannot join the collusion any more. However, this method produces high misjudgement rate of good nodes as it cannot differentiate the dropping behavior of malicious nodes and normal nodes (due to successful delivery, full buffer or packet expiry).

Unlike previous works, SDBG can detect both individual and collusion attackers with high accuracy and low false positive rate without relying on centralized servers. Besides, SDBG can work with multiple routing protocols (not restricted to Prophet only).

2.3 Flooding Attack

2.3.1 Detecting flooding in wired network and MANET

Flooding is a classic attack that threatens traditional wired networks such as Internet. It can be used to launch Denial of Service (DoS) attack to disrupt network services for legitimate nodes. The general approach of attackers is flooding a server or host with connection requests that cannot be completed so that these requests eventually overrun the victim's memory buffer. When this buffer is full, the victim can no longer process genuine connection requests and become unresponsive to legitimate traffic. The goal of flooding attack in traditional wired network is to exhaust the resource of a target node. In wired network, centralized routers are deployed to monitor the network traffic and filter malicious flooding messages [54–56] or impose the limit on the traffic [57,58]. These schemes cannot be applied to ad-hoc networks which operate without infrastructure support.

MANET is an infrastructure-less network of mobile devices connected by wireless medium. MANET routing protocols like AODV [24], DSR [25] are normally designed under the assumption of well-connected network (rather than mobile DTN). In these routing protocols, when a node wants to send messages, it broadcasts a Route Request (RREQ) message to find the route to the destination and the neighboring nodes help forward the RREQ. Attackers can disrupt the network by flooding unnecessary RREQs. Researchers have suggested schemes [59–62] to detect flooding attacks in MANET. In [59–61], nodes monitor neighbors' rate of sending RREQ and accordingly lower the forwarding priority for high-rate inquiries or blacklist the source of high-rate inquiries. Guo et. al. [62] design features of RREQ flows that characterize flooding attack and use them to detect the misbehavior. These schemes only work with MANET's routing protocol or require flow knowledge which is hard to collect in DTN.

2.3.2 Detecting flooding in DTN

To increase the delivery probability, the routing protocols in DTN often allow packets to be replicated and forwarded in multiple paths to the destination. Thus, the flooding attack in DTN could exhaust the resource of the whole network. [63] addresses the DoS attack in which attackers exhaust the memory and computational resources of other nodes due to the expensive strong authentication process. The scheme depends on a hierarchical infrastructure to filter invalid messages from accessing the target DTN. The network is grouped based on regions and each regional DTN is controlled by a security gateway and several security-aware nodes with higher power and resources than normal nodes. The gateway is responsible for filtering of incoming and outgoing messages between regional DTNs while security-aware nodes verify the messages they receive within the regional DTN. The authentication is implemented in two phases. The first phase is light authentication using a light-weight and hard to forge cookie which is constructed from symmetrical cryptography and hash functions. If this phase is successful, the stronger authentication (e.g. using asymmetrical cryptography) will be triggered. The scheme can efficiently prevent external attackers from flooding bogus messages, but could not detect internal attackers.

In [64], no authentication method is assumed, making flooding attack more severe. Specifically, any attacker can spoof as other identities to play as the source node or the relay node of the bogus message, making it harder to trace the real adversary identity. Without authentication service, [64] proposes a queueing mechanism to mitigate the effect of flooding attacks in Prophet-based DTN. It is observed that when a node receives too many messages from a particular seldom-encountered node, these messages are highly suspected as bogus and therefore should be removed first when the node's storage is full. Each node calculates the queue priority for each message based on the encounter frequency with the message's relay node and the message size. The lower the

queue priority, the sooner the message is dropped. The simulation results show that in the presence of flooding attacks, this queueing mechanism provides better delivery performance than other queueing mechanisms such as first in first out, most forwarded message first, etc. However, the scheme is not general enough to be applied to all DTN routing protocols nor can it eliminate the attacking source.

The resource misuse detection scheme [65] employs rate-limit in combination with a monitor gateway. The rate-limit policy restricts the number of packets generated by a node per time interval. When a mobile node encounters the gateway, it reports to the gateway the packets kept in its storage. Based on those reports, the gateway maintains the count of packets generated by a node in each time interval and the count of copies per packet forwarded by the packet's source node. The violation is detected by the gateway if the counts exceed the upper bounds defined in the rate-limit policy. The detection method is enhanced with probabilistic approach to reduce the detection latency. The scheme requires a centralized gateway which might become the bottleneck and single point of attack in DTN.

Instead of using a centralized gateway for monitoring purpose, [66] proposes a distributed approach to detect if a node has violated its rate limits. Authentication service is assumed to be available so that external flooding attacker could be deterred easily. Each node registers with an authority a rate-limit based on its demand to obtain a certificate to prove its rate-limit to other nodes. When the nodes' demand increases or decreases, it can change its rate-limit by requesting the authority for a new certificate. Each node itself counts the number of packets or copies that it has sent and claims the counts to other nodes. The receiving nodes carry the claims when they move and make cross-checking if the carried claims are inconsistent when they contact. The claim structure ensures that each receiver should receive a different claimed count for packet copies or newly generated packets. However, as malicious nodes hide their excess rates

by claiming counts smaller than the real values, the pigeon-hole principle guarantees that the attackers must reuse the count for different encountered nodes, which can lead to detection.

The rate-limit policy is rigid since it strictly disallows legitimate users to have on-demand burst traffic which exceeds the rate-limit for a short period. Obtaining new rate-limit certificate can help but it is only practical if the connectivity to the centralized rate-limit authority is available and always on (like 3G, cellular). Besides, if nodes have unpredictable traffic rate and frequently change their rate-limits, the cost of changing rate-limit is high.

Different from these works, FDER is a fully distributed flood detection and regulation scheme, which is not only independent of the underlying routing protocols but also able to accommodate burst traffic for legitimate users.

2.4 Trust Management and Privacy Preservation in VANET

2.4.1 Privacy preservation

The most common authentication approach in network is public key infrastructure (PKI), in which a trusted authority issues each valid node with a pair of public and private keys and a certificate to ensure the correspondence between the public key and the key owner's identity. The node uses its private key to sign a message before sending it and other nodes can verify the message's signature with the sender's public key and certificate. Using the same certificate over time makes nodes vulnerable to attackers who eavesdrop packets from the network to identify the sender and to estimate its

location with an accuracy sufficient for tracking it. By collecting different messages of a node, attackers can obtain its location history, thereby inferring its sensitive information (such as home/work address, locations of interest, etc.). If a node stays anonymous by not revealing its certificate, its privacy is protected but other nodes cannot verify its messages.

Existing privacy schemes enable nodes to stay anonymous and authenticated at the same time. They can be categorized into three types:

1. Group signatures
2. Pseudonyms based on public key infrastructure (PKI)
3. Pseudonyms based on identity-based cryptography (IBC)

Group signatures

Group signature enables an entity of a group to produce a signature on behalf of the group and the signature can be verified with the group public key. Privacy is achieved as the signatures are not linkable to the signers, allowing signers be anonymous within the group. A group manager (GM) is required to manage the group: setting up group parameters, changing group key, managing group membership, tracing member's identifier.

In GSIS [86], a centralized membership manager (MM) plays as GM to issue private keys and group public keys to the vehicles. Messages sent by vehicles are signed with the group signature to protect their privacy. When a vehicle's real identity needs to be resolved for liability investigation, MM looks up its database for the match between the secret key and the identity. To revoke a vehicle, MM needs to generate a new group key and derive new private key for each unrevoked group member. Thus the revocation

overhead of GSIS is huge for a large group of vehicles.

[87] proposes that groups are formed on the way as vehicles move and each Road Side Unit (RSU) plays as GM of the vehicles within its communication range. Vehicles request new private member keys when they enter the group or when their existing keys expire. Vehicles use their own private keys to sign their messages so that other vehicles in the same group can verify the messages. In case of liability investigation, the identity of the message originator could be revealed by RSU. However, if RSU is compromised, vehicles' privacy might be violated. Besides, for the vicinity that is not installed with enough RSUs, group signature authentication is not always available. The group management overhead is also huge since VANET is highly dynamic.

To resolve the issue of the revocation overhead, [88] proposes an efficient distributed key management (DKM) for group signature in VANET. The whole domain of VANET is divided into several sub-regions, each is managed by a regional GM. Each vehicle updates its group secret key periodically from the GM corresponding to the region where it stays. Since the group public key issued by each GM is only valid in its region, a revoked member is notified in a small region instead of the whole domain. Thus, the size of the revocation list is decreased. However, DKM restricts the group signature authentication within a region only, making it impossible to authenticate messages from different regions.

PKI-based pseudonyms

PKI-based pseudonym is a valid certificate stripped of any sensitive information (like nodes' real ID) and used for authentication. Each node uses one pseudonym at a time and signs its message with the corresponding private key. Receivers authenticate the message with the sender's pseudonym without the need to identify the sender.

Since the certificate is valid, the authentication is ensured. Privacy is achieved since the pseudonyms are uncorrelated to one another. Existing works discuss different aspects of PKI-based pseudonyms such as pseudonym issuance [81–83], pseudonym change [89–93].

[81] proposes an infrastructure including a certificate authority (CA) to issue long-term certificates and a pseudonym provider (PP) to issue short-term pseudonyms. Only nodes certified by CA can acquire new pseudonyms from PP. Before issuing pseudonyms, PP authenticates nodes using their long-term certificates and keeps the pseudonyms-to-identity mapping in case of liability investigation. Each node needs to own multiple pseudonyms so that it can change pseudonyms over time for privacy preservation. Thus it has to request PP for new pseudonyms periodically to be used for a period in the future, incurring high overhead of communication or storage.

To reduce the overhead, pseudonym self-issuance [82,83] is proposed. In PKI+ [82], each node first chooses a master key that is certified by CA and then self-generates pseudonyms from this key. These pseudonyms are CA-certified, so PP is unnecessary. In case of liability investigation, a node's identity can be resolved by having CA map between the master key and the identity. In [83], nodes utilize group signature to certify their self-issued pseudonyms. To authenticate a message, a node first verifies the group signature to determine if the sender's pseudonym is valid without learning the sender's identity. Afterwards, the message signature is verified from the sender's pseudonym.

Changing pseudonyms over time is not enough to ensure the privacy. For example, if there is a single node appearing in a region, even if that node changes pseudonym, the varied pseudonyms can still be linked to the same entity. Therefore, nodes need to change pseudonyms in proper manner or context to enhance their privacy efficiently. [89] suggests that vehicles change pseudonyms periodically at fixed time slot.

Besides, nodes can synchronize the change with their neighbors by broadcasting a message to notify their readiness to change. This scheme is vulnerable to tracking attack if the adversaries discover the fixed time slot duration. Thus, [90] proposes that vehicles change pseudonyms randomly to make it hard for adversaries to predict the next pseudonym change. However, tracking attack is still possible if the vehicle density is low. [91] suggests that social spots such as traffic light, parking, where many vehicles gather with the same initial speed of zero, are suitable places for vehicles to change pseudonyms. When the vehicles that share the same state and region update pseudonyms simultaneously, they create a mix zone that deters the tracking attack. This scheme is also susceptible to low density scenario. [92] optimizes the pseudonyms use by letting nodes change pseudonyms when the density exceeds a threshold. In AMOEBA [93], a vehicle remains silent after changing its pseudonym so that tracking this vehicle becomes more challenging. The trade-off of AMOEBA is that nodes cannot authenticate one another when they keep silent to protect their privacy.

IBC-based pseudonyms

Another type of pseudonym is based on IBC, in which a trusted server key generator center (KGC) derives private key for each node's identity value. To verify a message, nodes only need to know the sender's identifier without requiring the certificate. Thus, compared to PKI, IBC saves some bandwidth and time for exchanging certificate. Similar to PKI-based pseudonyms, IBC-based pseudonyms are a set of pseudo-identifier and key pairs.

[84] presents a network infrastructure to provide IBC-based pseudonyms. A trusted authority (TA) is responsible for (1) generating IBC system parameters and a master secret key, (2) generating and distributing a secret shared key with each base station (BS), (3) issuing each valid vehicle with a certificate, a public key and a private key (as

in PKI). IBC system parameters are published to both vehicles and BSs. Additionally, BSs get the master key from TA so that they can generate new pseudonyms for vehicles on behalf of TA. When a vehicle needs a new pseudonym, it sends a nearby BS its certificate in ciphertext and authenticates itself with BS. BS creates a new pseudo-identifier which contains the vehicle's real identifier encrypted by its symmetrical secret key and the BS's identifier. If liability investigation is required, the vehicle's real identifier could be traced back by decrypting its pseudo-identifier with the secret key of the issued BS. [84] incurs high overhead of changing pseudonyms.

[85] proposes an IBC-based scheme that enables nodes to authenticate one another without the need of revealing their identities. Therefore, vehicles are not required to change pseudonyms. Each node only needs its identity and private keys to verify if the other node's secret keys and its own keys are generated from the same system's master key. However, the verification requires each vehicle to be equipped with a tamper-proof hardware.

These schemes only focus on improving the functionalities of group signatures and pseudonyms to ensure the privacy and authentication among registered vehicles. However, they do not consider how the use of pseudonyms or group signatures influences the trust management of entities and data.

2.4.2 Trust management

There are two approaches to check the credibility of data:

1. Objective evaluation: data is verified against practical and intuitive models such as physics rules, safety rules and statistical properties.
2. Subjective evaluation: nodes judge the data by fusing the neighbors' opinions

towards the data, in consideration of entity trust.

Objective evaluation

[70–72] focus on evaluating beacon messages, which contain the nodes' information of locations and speeds, by checking the validity of the range of the parameters, nodes' relative positions, nodes' mobility, etc. Meanwhile, [73–75] focus on detecting false reports of road events by defining the models of nodes' responses to the event and verifying if the reporting nodes follow the models.

VEBAS [70] is a framework for vehicles to analyze if other vehicles' beacons are valid and update their reputations according to the analysis results. Specifically, a vehicle evaluates a received beacon as valid if the average velocity, acceleration and heading of the vehicle are valid and there is evidence that the beacon's originator has moved consistently for a certain period of time and for a minimum distance. Furthermore, the beacon is evaluated invalid if (1) the location specified in the beacon is too far from the verifier's location, or (2) if the location is not found on the road map, or (3) if the location in the first received beacon is too near to the verifier (i.e. the beacon's originator suddenly appears near the verifier). If the beacon is valid, the verifying vehicle increases the trustworthiness towards the beacon's originator vehicle and vice versa. The local trust values can be exchanged among vehicles to build the reputation system for the network.

[71] proposes a plausibility model to verify if the vehicles' locations and movements are valid. The plausibility model assumes that only one single physical vehicle can occupy a certain space at certain time. The space occupied by a vehicle is modelled as a rectangle constructed from the location and heading direction specified in its beacon, the vehicle's dimensions (length and width) and the safety clearance distance

from surrounding vehicles. If adjacent vehicles have their occupied space overlap, the location of at least one vehicle is concluded as invalid.

[72] proposes a verification framework based on Kalman filter to verify the mobility of neighboring vehicles even under privacy considerations (i.e. changing pseudonyms). Kalman filter is used to estimate a vehicle's future movements from its current mobility information and build a tracker profile for each vehicle. The framework first verifies the beacon against several thresholds as in [70] and then checks if this beacon is consistent with the tracker profile of the originator vehicle. If the originator does not change pseudonym, the tracker profile could be easily recognized by the pseudonym. Otherwise, the framework searches for an appropriate tracker profile by comparing the received and predicted mobility data of all existing trackers. Though this framework enables both privacy and trust management, the data verification is only limited to mobility data.

[73] proposes MDS to detect the malicious nodes that send the false alert of a non-existing post-crash notification. If the alert is authentic, the alerting vehicle should take proper actions in order to avoid the crash. To verify the credibility of the alert, the locations of the alerting vehicle are monitored from the alert time until that vehicle passes the crash site. The alerting vehicle's expected trajectory during this period of time is calculated based on its initial lane and the lane of the crash. The alert is detected as false if the difference between the two trajectories is above a certain threshold.

[74] proposes a fox-hole model as the unsafe region around the location of reported road danger that vehicles should avoid or run slow nearby. By observing the response of alerting vehicles to the fox hole, the alert could be evaluated as true or false. [75] proposes another response model of vehicles towards different events that considers multiple factors such as heading direction, lane and speed of vehicles. The event is

evaluated as false if the vehicles do not follow the response model.

The data verification frameworks proposed in [70–72] are limited to vehicles’ mobility only. [73–75] can verify event data using the trajectory information of vehicles for a period of time, which might be hard to achieve under the privacy context of pseudonym change. Besides, the objective evaluation presented in [70–75] could only be possible if the vehicles are near enough to the verified target to listen to its beacon.

Subjective evaluation

[76] defines a sociological trust model for VANET including various types of trust: situational trust, dispositional trust and system trust. Situational trust models the application’s ability to adapt trust threshold in critical situations, or situations with sparse information. Dispositional trust models the trust based on the information plausibility test. System trust depends on the strength of the trust evaluation method used in the network. Though [76] proposes that a belief formation process is required to evaluate the data based on defined trust factors, it does not formalize and specify how to do it.

[77] introduces a trust model that integrates IBC-based entity trust and email-based social trust. The entity trust provides basic security functions such as vehicle authentication, data integrity, and confidentiality. The social trust provides additional trustworthiness information towards the data generated by other vehicles. Each user uses its email address as identity. When users exchange data, they can also exchange identities to obtain email-based social trust and to establish a group key for securing the communication among users. To obtain the social trust, each user runs an email trust checking agent (TCA) software on its device that sends trust checking requests and ranks the peers’ trustworthiness based on their email accounts. To secure the group

communication, each user constructs an access control tree such that only receivers satisfying the access control tree can decrypt the message. Though the email-based trust might be useful, obtaining it requires the connectivity to email service and may incur long delay due to network latencies.

[78] proposes a multi-faceted trust management model that allows a vehicle to evaluate data by actively inquiring other entities for their opinions towards an event. To reduce the communication overhead, the number of received reports can be restricted by giving higher priority to reports generated by entities with authority role and higher reputation. The vehicle makes a final evaluation of the data by executing majority voting on the received opinions on account of the reporting entities' role-based trust and experience-based trust and dynamic factors of location and time of the event report. Specifically, reports from nodes closer to the event's location and reports received more recently are more trustworthy. The evaluation result is in turn used to update the experience-based trust for entities. The scheme does not consider how to manage the entity trust under the privacy context of pseudonym change.

[79] proposes a framework for data-centric trust establishment in which the trust in each individual piece of data is computed first and the trust in various data are combined using a decision logic. There are four decision logics to aggregate and evaluate evidences about an event: most trusted report, majority voting, Bayesian inference, Dempster-Shafer theory. Though the framework utilizes the entity trust to compute the weight for the data before feeding the weight to the decision logic, [79] does not specify how to achieve this entity trust.

[80] studies the information cascading and oversampling problems in the simple voting scheme used to integrate data in VANET. The opinions about an event given by certain nodes can be too overwhelming, leading to incorrect decision towards the event. To resolve this issue, [80] also proposes a new voting scheme in which each vehicle

has different voting weight according to its distance from the event. Specifically, the vehicle closer to the event is assigned with higher weight.

2.4.3 Trust management in privacy context

[94] proposes a message-linkable group signature scheme to support the verification of data. The signatures of a vehicle on different messages are unlinkable, allowing it to be authenticated anonymously. However, the signatures of a vehicle on the same message are linkable, allowing the detection of vehicles that maliciously sign a message for multiple times to increase the data's credibility. Each vehicle generates a secret key, registers the corresponding public key with the trace manager and obtains the group public key from the registration manager. After that, each vehicle can endorse a message it believes with group signature. If liability is requested, the trace manager will resolve the vehicle's real identity. However, the scheme incurs high overhead of revocation as in [86].

[95] suggests using group signature for messages and feedbacks to protect privacy. A central server is required to receive nodes' feedbacks towards other nodes, fuse these feedbacks and issue nodes with "reputation certificates" accordingly. The availability of such central server in VANET context is impractical.

[96] proposes SBAP to enable both trust management and privacy protection. SBAP allows a node, say B , to temporarily track varying identifiers of neighboring vehicle, say A , thus helping B to observe and update the peer trust towards A . In SBAP, A self-generates a pseudonym from a secret key and a random number. By changing the random seed, A can change its pseudonym when necessary to protect its privacy. To track A , B needs to acquire A 's secret key by sending a request that contains B 's certificate and its signature. Once A authenticates B as a valid user, A

encrypts its own certificate and secret key with B 's public key and sends it as a reply to B . After decrypting the reply, B can authenticate A with A 's certificate and track A with A 's secret key.

SBAP is lightweight and simple but it has several shortcomings. First, to request A for tracking, B needs to reveal its certificate in plain text, making B vulnerable to the privacy threat. Note that B cannot encrypt its certificate since B does not know A 's certificate when A uses pseudonym. The pseudonym scheme in SBAP does not provide encryption function either. Second, the functions of authentication and encryption are coupled with the tracking function as B does not know A 's certificate unless it requests to track A . Thus, it is very awkward to apply SBAP for applications that require authentication or encryption only. Third, by changing the current secret key, A disables all neighboring nodes from tracking it. A cannot selectively disable the tracking of certain nodes while remaining to be tracked by other nodes. They can only continue to track A by repeating the interactive request process.

Unlike previous works that only focus on trust management or privacy, the framework proposed by the authors address both issues with reasonable trade-off. Compared to SBAP that also provides linkability, ALRS is more secure and flexible. Better privacy against external eavesdroppers is offered due to the use of IBC-based pseudonyms to encrypt nodes' identities before they are sent out. ALRS separates the functions of authentication/encryption (using pseudonyms) and linkability (using linkability protocol). ALRS supports three levels of linkability while SBAP only provides two levels of identity recognition and revocation. ALRS allows selective linkability while SBAP cannot.

Chapter 3

Detecting and Localizing Wormhole Attack in DTN

This chapter presents SNDW (Statistical Neighbor-count-based Detection of Wormhole attack) to detect wormhole attack in DTN. SNDW uses infrastructure nodes to monitor if the neighbor counts in their vicinities experience an abnormal increase compared to the normal count, which is the symptom to detect the wormhole presence. The simulation results show that SNDW detects wormhole in DTN with high accuracy and low false positive.

3.1 System Model

This section discusses how to set up the infrastructure system that is responsible for detecting the wormhole attack in DTN. The author illustrates the setup in an example network with the bounded area of 1kmx1km and the transmission range of 100m in Fig. 3.1(a). It is assumed that the wormhole endpoints are possibly located in the blue shaded area only, as in Fig. 3.1(a). This is reasonable because wormhole nodes positioned too near the edge with very low density can hardly create significant influence on the network. The affected network area is divided into 16 small squares, each has side length of 200m. The helper infrastructure consists of 16 nodes set out in the area so that each node is responsible for monitoring under its sub-area. Infrastructure node listens to beacon messages broadcast by the mobile nodes and collect neighbor counts over time. If the infrastructure node is within the transmission range of a wormhole endpoint, it will receive more beacon messages than normal, corresponding to a significant increase in neighbor count.

In order to cover the entire square that an infrastructure node is in charge of, it follows a scheduled movement pattern as shown in Fig. 3.1(b), cyclically visiting all the vertices of the small square in the order of A, B, C, D . At each vertex, it stops for some seconds (called pause period) for monitoring before proceeding to the next vertex. The union of the four circles represents the detection range and they show that the infrastructure node can cover the entire area as long as the communication range of mobile nodes is larger than $200 \div 2\sqrt{2} = 70.7\text{m}$. Thus this schedule works for the given transmission range of 100m.

In general, the infrastructure setup could be determined as follows. First is to determine the network area that could be affected seriously if the wormhole attack is present. Normally, this area should be in the centre of the network area where multiple

Figure 3.1: (a) Distribution of helper nodes. (b) Movement of helper nodes.

nodes often pass by. Second is to divide the affected network area into small squares, each has size $d \times d$ where $d \leq 2\sqrt{2}r$, where r is the transmission range of nodes in the network. In this way, when each square is assigned to an infrastructure helper node and the helper node follows the monitor schedule as in Fig. 3.1(b), the helper could cover the whole assigned area.

It is also assumed that there is no wormhole attack at the beginning, but it is established by the adversary during the operation of the network. This aims to allow the infrastructure to have base knowledge of the normal network density so that it can detect anomaly when wormhole appears.

3.2 SNDW

3.2.1 Detecting wormhole

Fig. 3.2 displays a typical statistical situation where the neighbor counts increase significantly due to wormhole attack. The upper plot shows the raw data of neighbor

counts which suggest a potential increasing trend when a wormhole appears. The raw data is smoothened by averaging neighbor counts over a time window of 50 seconds as shown in the lower plot. This pre-processed data provides better evidence for the detection algorithm.

Figure 3.2: Example of neighbor count statistic.

The detection process of SNDW includes two phases: training and testing. As mentioned above, it is assumed that wormhole attack is not present in the network from the beginning. Training yields the average of maximum neighbor count over a period in which it is known that there is no attack for sure. This mean count represents the typical and legitimate density in the sub-area. Testing involves checking if the ratio between the current neighbor count and the mean count exceeds a threshold τ . If the threshold is exceeded frequently for a time interval, wormhole attack is declared as present. Algorithm 1 illustrates the two-phase wormhole detection procedure.

Algorithm 1 Wormhole Detection

T_O : *Initial Time*
 N : *Time Period, with no wormhole since T_0*
 $getWindow(t, M)$: *Return array of M consecutive neighbor counts from time t*
 τ : *detectionthresholdTime Period, with no wormhole*

{training}
 $Window_{train} = getWindow(T_0, N) = \{D_1, \dots, D_N\}$
 $\bar{C} = \frac{1}{N} \times \sum_{i=1}^N D_i$

{testing}
 C_1 = *current neighbor counts*
 t = *current time*
if $C_1/\bar{C} \geq \tau$ **then**
 $Window_{test} = getWindow(t, T) = \{C_1, \dots, C_T\}$
 violation = 0
 for $i \leftarrow 1$ **to** T **do**
 if $C_i/\bar{C} \geq \tau$ **then**
 violation ++
 end if
 end for
 if *violation* $\geq T/2$ **then**
 wormhole = *TRUE*
 else
 wormhole = *FALSE*
 end if
end if

3.2.2 Localizing wormhole

After detecting wormhole attack in its own area, the infrastructure node will notify its nearby neighbors. Fig. 3.3 shows three different possibilities of wormhole endpoint location relative to the coverage range of the helper infrastructure: (i) in the overlapping sub-areas of 4 infrastructure nodes; (ii) in the overlapping sub-areas of 2 infrastructure nodes; (iii) in only 1 sub-area. In Fig. 3.3(a), all the four infrastructure helpers in the four small squares could detect the presence of wormhole attack. As a result, an wormhole endpoint should be located in the intersection area of their respective

Figure 3.3: Possible wormhole location corresponding to wormhole declaration by: (a) four helpers, (b) two helpers, (c) one helper

monitor ranges which is shaded in red color. Similarly in Figs. 3.3(b) and 3.3(c), the area shaded in red suggests the potential location of a wormhole attacker. Therefore the wormhole detection information from the infrastructure nodes can be collated to narrow down the suspected range of the wormhole placement, thereby pinpointing its near-exact location.

3.2.3 Discussion on statistical approaches to detect wormhole

In the literature, the researchers have already proposed wormhole detection methods based on statistical neighbor counts analysis [36,37]. However these methods are targeted at WSN in which the node density is very high, thus the appearance of wormhole will create a significant influence on the neighbor count data and the suggested sta-

tistical tests can successfully detect the change. In sparse DTN, wormhole can only increase the number of neighbors of the affected nodes marginally. As a result, there is not much change in the distribution of neighbor counts over the network for the cases of with and without wormhole, as shown in Fig. 3.4. Thus [36] will fail in the detection. The subtle change in neighbor counts also makes it impossible for each mobile sensor to differentiate between a higher-density area from a lower-density area which is under wormhole attack as in [37]. Fig. 3.5 illustrates the neighbor count of a single node over time where the marks indicate the wormhole occurrences. It is observed that the statistics show no difference between when a node travels close to the wormhole and when it does not.

Figure 3.4: Distribution of neighbor counts in DTN with and without wormhole.

To overcome the mentioned challenges of statistical approaches in DTN context, the author proposes the semi-distributed scheme using helper infrastructure to collect and analyze the data for the network. This idea can work because each helper is dedicated to monitor a small area where the density can be assumed to be static over time if the mobile model does not change. Therefore, even with a subtle increase in the neighbor counts, the responsible helper is still able to detect the abnormal increasing trend in the affected sub-area.

Figure 3.5: Statistical change of neighbor counts in DTN.

3.3 Performance Evaluation

The simulation is implemented in NS3 [27] with the following settings: 40 mobile nodes with speed of $[0, 20]$ m/s and transmission range of 100 m. Nodes follow the random waypoint mobility model. The network area is 1 km x 1 km and the layout of the helper infrastructure is as per that presented in Section 3.1. The simulation time is 3200 s, in which the first half is not attacked by wormhole. A pair of wormhole endpoints is placed randomly and fixed during the simulation. There are two parameters of SNDW as described in Section 3.1: the helpers' pause time ρ and the testing count threshold τ . The default value of ρ is 100 s and the default value of τ is 2. This general setting is applied to all the following experiments except when other settings are specified. For each experiment, the simulation runs for 20 times with random seeds and the average of the measured metrics are recorded and presented.

The performance is measured using the following metrics:

1. *Detection time*: average time taken to declare the presence of wormhole from the start of the attack

Table 3.1: Performance of SNDW vs prohibited-topology method

Algorithm	SNDW	Prohibited Topology Method
Detection time(s)	134.5	357
Detection rate	1.0	1.0
False positive rate	0.03	0.30

2. *Detection rate*: ratio between number of runs that can successfully detect wormhole over total number of runs
3. *Detection false positive*: average ratio between the number of wrong declarations over the total number of declarations.

3.3.1 Comparing SNDW with prohibited-topology method

Table 3.1 shows the performance result of SNDW and the prohibited topology method [42]. In comparison with [42], SNDW is equivalent in detection accuracy but it can detect much faster and with lower false positive. It is noted that the performance of wormhole localization depends on the detection performance. The higher the detection rate and the lower the detection false positive, the higher the localization accuracy and the lower the localization false positive.

The prohibited topology method assumes the model of unit disk graph (UDG) and therefore the neighbor relationship can be judged based on the distance between a pair of nodes. However, in reality, the fast movement of the mobile nodes and the propagation loss or collision of network MAC layer can affect the maintenance of neighbor list, leading to an increase in false positive rate. The proposed scheme is based on the number of neighbors only (not the identity of neighbors), therefore it can work even if there is propagation loss or when the mobile nodes move at high speed. As in Fig. 2.3, the wormhole can be detected successfully only if the position of C is within the half-range of one wormhole endpoint and the positions of D and E must be

within the range of the other endpoint. In sparse DTN where neighbors are scarce, the mentioned positional condition can be hard to achieve, leading to long delay in detecting the attack. Between the two schemes, SNDW is equally effective for higher speed networks (with speeds of up to 20 m/s) which means it is also applicable to vehicular DTNs. The disadvantage of SNDW is the requirement of additional infrastructure. However, it is noted that the functionality of the infrastructure helper can actually be incorporated into existing deployed infrastructure mesh network and hence there is no need to deploy additional helpers. SNDW does not consume much computation capability and memory of helpers because it only requires simple functions (receiving beacon broadcast, updating neighbor counts and analyzing data) and neighbor count information does not occupy much space.

3.3.2 Varying network parameters

The network parameters such as number of nodes, transmission range are varied in two scenarios:

1. The node density is fixed at 40 nodes. The transmission range is varied from 70 m to 130 m.
2. The transmission range is fixed at 100 m. The number of nodes is varied from 20 to 50.

Fig. 3.6 and 3.7 illustrate the relation between the varied parameters and the performance metrics. It is observed that the detection performance is independent of these network parameters. This can be explained as follows. SNDW is based on the increasing trend in neighbor count over a small area. Therefore as long as the sub-area's neighbor count is stable over time, SNDW can detect the ratio difference before

Figure 3.6: Performance of SNDW under varying transmission range.

Figure 3.7: Performance of SNDW under varying number of nodes.

and after the wormhole presence regardless of the absolute value of neighbor count in the area.

Meanwhile, the prohibited-topology method [42] is dependent on the above network parameters. When the transmission range increases or the number of node increases, each node has more neighbors. [42] requires the involvement of three nodes falling within the range of the wormhole in order to detect the abnormality. Thus more neighbors mean higher possibility of forming the forbidden structure, leading to faster detection and higher accuracy.

3.3.3 Varying method parameters

SNDW parameters are varied with the following values: $(\rho, \tau) = (100, 2); (200, 2), (200, 1.8)$.

Fig. 3.8 illustrates the performance results of SNDW when the method parameters are varied. It is observed that higher pause time ρ leads to slower detection time, higher accuracy and lower false positive. This is because the longer the helper stays at one place, it can collect enough neighbor count evidence in that location to declare the wormhole with more confidence and precision. However, if the wormhole-attacked area is the last one to be visited in the cyclic monitor, the detection time is higher since it is delayed more as it spends time in its previous locations.

It is also seen that smaller threshold τ leads to quicker detection time, higher detection accuracy and much higher false positive at the same time. This is because the smaller value of threshold makes the method more sensitive to the change in neighbor count.

Figure 3.8: Performance of SNDW under varying method parameters: (a) Detection time. (b) Detection accuracy. (c) False positive.

3.4 Conclusion

The author proposes SNDW which is a semi-distributed statistical method to detect wormhole in DTN. The analysis shows that SNDW works in DTN while previous statistical approaches fail in such sparse network scenario. Compared to the prohibited topology method which also aims at defending against wormhole in DTN, SNDW achieves higher detection accuracy, faster detection time and lower false positive. The performance of SNDW is independent of the network density and node transmission range while there is a trade-off in performance when varying parameters namely node pause time and detection threshold.

Chapter 4

Detecting Blackhole and Greyhole Attacks in DTN

This chapter presents SDBG (Statistical Detection of Blackhole and Greyhole attack) that addresses both individual and collusion blackhole and greyhole attacks in DTN. SDBG requires nodes to exchange their ER histories, based on which other nodes can evaluate their forwarding behaviors. The forwarding ratio metrics, which can be inferred from ER history, are computed to detect individual attackers. Malicious nodes might avoid being detected by colluding to manipulate their forwarding ratio metrics. To continuously drop messages and promote the metrics at the same time, attackers need to create fake ERs frequently and with high forged numbers of sent messages. The abnormal pattern of appearance frequency and number of sent messages in fake encounters is exploited to detect colluding attackers. Extensive simulation shows that SDBG can detect both individual and collusion attacks with high accuracy and low false positive under various attacker settings.

4.1 System Model

This section introduces the system setting under DTN scenario and describe the behavior model of normal nodes and individual attackers.

4.1.1 Mobile node model

The author considers the DTN mobile network scenario which involves mobile nodes communicating in ad-hoc mode with wireless technology such as WiFi. The author also assumes a trusted authority with the right to assign each node a unique identifier and a pair of public and private keys. Nodes are assumed to know the public keys of one another so that they can authenticate messages signed by others.

Under the above settings, the author models the general behaviors of nodes as follows. When two nodes encounter and exchange messages, each of them generates an Encounter Record (ER) and stores it in its own storage. The ER includes the identities of two nodes, the ER sequence numbers assigned by them, the encounter timestamp and the lists of sent and received messages between the two parties and their signatures.

To ensure that nodes agree on the encounter timestamp in ER, the network is loosely time synchronized. This means that the time is divided into slots of equal duration. At any one time, any two nodes have their local clock times falling in the same time slot. Given that in DTN which is often characterized by sparse nodes and infrequent encounters, the inter-contact time is usually in minutes if not hours. The time slot can be set to any length that is suitable for the DTN concerned. The time slot can therefore be in duration of minutes.

Suppose the two nodes i and j with respective identifiers ID_i and ID_j meet each other. The record ER_i generated by node i is formatted as follows:

$$ER_i = \langle ID_i, ID_j, sn_i, sn_j, t, SL_i, RL_i \rangle \quad MR_m = \langle ID_m, SRC_m, DST_m \rangle$$

$$SL_i = \{MR_m \mid i \text{ sends message } m \text{ to } j\}$$

$$RL_i = \{MR_m \mid i \text{ gets message } m \text{ from } j\}$$

$$ER_i^* = ER_i, sig_i, sig_j$$

where sn_i, sn_j are the sequence numbers assigned to the ER by node i and j respectively; t is the encounter time (in units of time slots); sig_i, sig_j are their signatures over the content of ER_i using their respective private keys. SL_i and RL_i denote the lists of message records that node i sends to and receives from node j respectively. Each message record (MR) contains 3 fields: the identifier of the message, the identifier of the source node who generates the message and the identifier of the destination node to whom the message is destined. A node assigns its new ER with the sequence number of its latest ER incremented by 1.

Due to possible storage limitation, each node only keeps a window of w most recent encounter records and presents them to the neighbors. Each node maintains the meeting list (ML) to store the summary information for other nodes based on their encounter histories. For each node j that i previously encountered, i creates a meeting summary (MS) MS_j^i for j which includes the identification of j , the sequence number and timestamp of the latest encounter record of node j and j 's trust reputation. Each entry MS_j^i is formatted as follows:

$$MS_j^i = \langle ID_j, latest_sn_j, latest_t_j, TR_j^i \rangle$$

where ID_j denotes the identifier of node j , $latest_sn_j$ and $latest_t_j$ are the sequence number and contact time of the latest encounter record of node j known to i . Node i can learn this information when it meets either node j directly or some other node k who has a recent encounter record with j . TR_j^i represents the most updated trust reputation of node j judged locally by node i . When two nodes first meet, they can

set the initial reputation TR_{init} for each other.

4.1.2 Individual packet-dropping attack model

Individual adversaries launch the attack by dropping messages and manipulating their own ER histories. An attacker first receives messages from other nodes but later drops them with a certain probability. Blackhole attacker drops all received messages (dropping probability = 100%) while greyhole attacker drops the messages partially (dropping probability lower than 100%). The dropping occurs even if the attacker still has enough buffer to store messages.

To conceal its dropping misbehavior, an attacker may manipulate its own ERs history to obtain an encounter record window beneficial for itself to present to other nodes. In Fig. 4.1, for example, the malicious node m has an actual series of encounter records A, B, C, D with respective $\langle \text{sequence number}, \text{timestamp} \rangle$ of $\langle 1, 10min \rangle$, $\langle 2, 20min \rangle$, $\langle 3, 30min \rangle$ and $\langle 4, 40min \rangle$. If the adversary wants to delete the unfavorable ER C with sequence number 3, it can use one of the following three manipulation strategies:

1. **Skipping ER sequence number.** m deletes the ER C and does not generate any other ER using the sequence number 3.
2. **Reusing ER sequence number.** m deletes the ER C and uses the sequence number 3 to generate another ER $E \langle 3, 50min \rangle$.
3. **Holding back unfavorable ERs.** m uses the same sequence number 3 to generate different ERs: $C \langle 3, 30min \rangle$ and $D \langle 3, 40min \rangle$. After achieving a favorable ER, it will update the sequence number to generate new ERs.

Figure 4.1: ER manipulation strategies of attackers.

4.2 Detecting Individual Attackers

Individual adversaries may have two misbehaviors as presented in Section 4.1.2. Both misbehaviors leave some traces that can be exploited to design the defense method. When manipulating ERs, attackers violate the consistency rules of creating encounter records. When dropping messages, malicious nodes only relay parts of the messages they receive and prefer sending messages for themselves to sending for other nodes. The author designs proper forwarding ratio metrics that reflect this observation to detect dropping.

The evaluation process is as follows. When two nodes i and j encounter each other, node j submits a window of latest w ERs for node i to assess its behavior and vice versa. Node i will check if node j manipulated the encounter records history or dropped messages. If node j is detected as committing either misbehavior, node i will punish it accordingly. Each node maintains a local black list which lists malicious nodes that it detects as blackhole or greyhole attackers.

4.2.1 ER manipulation misbehavior

According to the rule of creating ER, a series of consecutive well-behaved ERs has sequential sequence numbers. Besides, ER with higher sequence number has a bigger timestamp. However, adversaries using the first two strategies violate either rules as seen in Fig. 4.1. In the first strategy, the new ER history has the series of sequence numbers = 1, 2, 4 which is not sequential. In the second strategy, the new ER history has the series of time stamps = 10, 20, 50, 40 which is not monotonically increasing. When the manipulated ER window is presented to a neighbor node in later encounter, the misbehavior is easily detected.

The process of detecting strategies 1 and 2 is as follows. Upon encountering node j , node i obtains j 's encounter history window and examines if the records follow the sequence and timestamp order. Besides, node i can also cross-check each node k that j encountered within the window. The encounter record between j and k will have its sequence number sn assigned by k and timestamp t be checked against $latest_sn_k$ and $latest_t_k$ recorded in node i 's meeting list. If any node violates the rules, node i will blacklist them.

In the third strategy, the manipulated ERs history is A, B, D which abides by both sequence number and timestamp rules. Thus the neighbor cannot discover the manipulation at all if it depends on the ER window presented by m only. However, the misbehavior leads to the inconsistency that at least 2 different records C and D have been assigned with the same sequence number by m . Fig. 4.2 illustrates how a normal node k can discover a malicious node m who uses the same sequence number 3 to create ERs C and D with 2 nodes i and j . In Fig. 4.2(a), upon encountering i , k notes and records that m has assigned the sequence number 3 to an ER with i when it processes i 's ER history. If later on, k also encounters j , it will discover that m has

also assigned the sequence number 3 to the encounter with j . m is thus detected as malicious. Another variation of the detection process is shown in Fig. 4.2(b) where node m itself encounters node k . Upon meeting i and m and checking their submitted encounter history, node k can also detect that m has used the same sequence number 3 in both encounters with i and j .

(a) Example 1.

(b) Example 2.

Figure 4.2: Detecting inconsistent assignment of ER sequence number.

The author exploits this observation to detect the third strategy as follows. When each node i receives and processes an ER history of a neighbor j , it records the sequence number assignment by other nodes, not only by j but also by those encountering j . In this way, each node i has a global view of sequence number assignment over the network so that it can inspect if any sequence number inconsistency exists. Once detected, the malicious node committing the inconsistency is blacklisted. The process does not incur additional communication overhead and only requires a small storage of sequence number history.

4.2.2 Dropping misbehavior

Attackers might receive a lot of messages but only relay a small portion of them as the rest have been dropped intentionally. Among the messages sent out by selfish attackers, a large portion are messages generated by themselves as they drop others' messages but keep their own. Based on these observations, the author defines the metrics named relaying ratio RR and self-forwarding ratio SFR that can potentially reveal the dropping misbehavior as follows:

$$RR = \frac{N_{RS}}{N_{RNS}}$$

where N_{RS} is the total number of messages that j has received and has already relayed by forwarding to another node; N_{RNS} is the total number of messages that j has received as a relay (not the ultimate destination) but j has not sent out yet.

$$SFR = \frac{N_{self_send}}{N_{send}}$$

where N_{self_send} is the total number of messages that are generated and sent out by node j ; N_{send} is the total number of messages sent out (regardless of the message source). It is noted that N_{RS} , N_{RNS} , N_{self_send} , N_{send} are calculated over all the encounter records in node j 's ER window.

Due to the dropping misbehaviors, malicious nodes often have lower RR and higher SFR than well-behaving nodes. After obtaining the two metrics, node i compares them with predefined thresholds to take corresponding action towards updating the reputation for node j . If RR falls below the threshold Th_{RR} , the reputation of node j judged by node i will be decreased by γ . If SFR exceeds the threshold Th_{SFR} , the reputation of node j is further reduced by ρ . However, if both thresholds are not

violated, node j will have its reputation increased by λ . All the thresholds are chosen empirically using simulations.

The node whose reputation exceeds threshold Th_{good} are trusted and given higher priority to receive messages while those whose reputations fall below threshold Th_{mal} are blacklisted and isolated. Upon encountering a blacklisted node b , a normal node n will not exchange messages with it. However, node n still acquires the encounter record history of node b to obtain its measured SFR and RR . After collecting these metrics for some time and observing that the blacklisted node keeps behaving well, node n can allow node b back to its friend list but resets the initial trust reputation for b to be lower than TR_{init} . The purpose of this forgiving policy is to give the malicious nodes a second chance to change their behavior. Besides, it prevents the network from totally isolating the normal nodes who have been judged wrongly as malicious nodes.

An attacker that purely drops messages can be detected as presented above. An attacker that not only drops messages but also individually manipulates ERs might achieve a favorable ER history window to avoid violating the thresholds of RR and SFR metrics. However, the manipulation misbehavior can be detected by the scheme in Section 4.2.1.

4.3 Detecting Collusion Attackers

An advanced technique of blackhole and greyhole attackers is that they collude with one another to conceal their misbehaviors. In the presence of collusion, the detection of individual attack suggested in Section 4.2 is not effective any more. In this section, the author first presents the collusion attack model to illustrate how colluders cooperate to avoid the detection. The author then explains the proposed scheme to deal with the collusion. Table 4.1 summarizes the notation of the parameters.

Table 4.1: Notation of SDBG's parameters.

Given a node j and its encounter window ERW which includes several recent encounter records, the following parameters are defined for node j and the window ERW	
N_{RS}	Total number of messages that node j has received and has already relayed within its encounter window
N_{RNS}	Total number of messages that node j has received as a relay but has not sent out yet within its encounter window
N_{self_send}	Total number of messages that are generated and sent out by node j within an encounter window
N_{send}	Total number of messages that are sent out by node j within an encounter window
RR	Relaying ratio, defined by the ratio of N_{RS} over N_{RNS}
SFR	Self-forwarding ratio, defined by the ratio of N_{self_send} over N_{send}
Th_{RR}	Threshold of relaying ratio. If node j 's relaying ratio is below this threshold, its reputation is punished
Th_{SFR}	Threshold of self-forwarding ratio. If node j 's self-forwarding ratio is above this threshold, its reputation is punished

4.3.1 Collusion packet-dropping attack model

In collusion attack, adversaries cooperate so that they can counter against the individual detection scheme proposed in Section 4.2 and avoid being detected as individual attackers. It is assumed that colluding attackers consistently misbehave over time without switching their behavior between malicious and normal states. The behavior-switching attack will be discussed later in Section 4.4.5.

The strategy for an adversary is creating fake encounter records with its colluders to manipulate its own metrics RR and SFR . Colluding attackers are assumed to know the private keys of one another and freely use them to sign fake encounters to make them appear authentic. When the malicious nodes need to forge encounters, they need to determine the parameters for the fake encounters such as number of sent and received messages, encountered peer ID.

Suppose a malicious node m currently has its ER window denoted as $ERW =$

$\{ER_1, \dots, ER_w\}$. Node m 's corresponding parameters N_{RNS} , N_{RS} , N_{send} , N_{self_send} , RR , SFR can be inferred from ERW as shown in Section 4.2. A malicious message-dropping attacker is likely to have its metrics RR and/or SFR violate the thresholds:

$$RR = \frac{N_{RS}}{N_{RNS}} < Th_{RR}$$

$$SFR = \frac{N_{self_send}}{N_{send}} > Th_{SFR}$$

The violation of either rule causes m 's reputation to be punished by others. To avoid punishment, m needs to forge an ER so that the dropped portion is compensated and the new metrics over all its manipulated ER history are out of the abnormal range defined by the thresholds.

Denote the newly forged encounter record's parameters as: number of messages sent for itself n_{self_send} , number of sent messages n_{send} , number of received messages n_{recv} . Suppose this fake ER makes the total number of messages received but not relayed (over the new encounter record window) increase by n_{RNS} and the number of messages received and relayed increase by n_{RS} . To hide the violating metrics, m has to choose the parameters such that:

$$RR' = \frac{N_{RS} + n_{RS}}{N_{RNS} + n_{RNS}} \geq Th_{RR} \quad (4.1)$$

$$SFR' = \frac{N_{self_send} + n_{self_send}}{N_{send} + n_{send}} \leq Th_{SFR} \quad (4.2)$$

Attacker first needs to choose n_{RS} , n_{RNS} , n_{self_send} and n_{send} for the forged record to satisfy the conditions of Eq. 4.1 and 4.2; then it creates fake sent and received lists, SL and RL , to fit those chosen parameters.

Next, the attacker needs to choose the encounter peer node for the forged record

if it has multiple colluders. It can choose the fake encountered peer as the one that it creates the least number of forged encounters in its ER history window. In this way, the misbehaving node can minimize the encounter frequency with each colluder and avoid raising suspicion from the network. Finally, m can insert the signatures of itself and the chosen colluder over the fake encounter record.

The author will show that, to satisfy the conditions of Eq. 4.1 and 4.2, the number of sent messages in forged record n_{send} must be lower bounded. First it is assumed that node m simply sets n_{self_send} to 0 and n_{recv} to 0. As a result, $n_{RNS} = 0$ and $n_{RS} \leq n_{send}$. The fake metrics RR' and SFR' are formulated as:

$$RR' = \frac{N_{RS} + n_{RS}}{N_{RNS}} \geq Th_{RR}$$

$$SFR' = \frac{N_{self_send}}{N_{send} + n_{send}} \leq Th_{SFR}$$

This is equivalent to:

$$n_{RS} \geq Th_{RR} \times N_{RNS} - N_{RS}$$

$$n_{send} \geq N_{self_send} / Th_{SFR} - N_{send}$$

As $n_{send} \geq n_{RS}$, the author obtains the lower bound for n_{send}

$$n_{send} \geq \max(Th_{RR} \times N_{RNS} - N_{RS}, N_{self_send} / Th_{SFR} - N_{send}) \quad (4.3)$$

If m sets n_{self_send} and n_{recv} larger than 0, n_{send} even has to be larger.

4.3.2 Analysis of collusion behavior

The collusion attack presented above will nullify the individual detection scheme proposed in Section 4.2. The author needs to explore the measures that possibly reveal the collusion behavior. If the attacker keeps dropping received messages, its metrics RR and SFR will be degraded continuously. To avoid being detected, it needs to keep promoting the metrics by creating fake encounters persistently over time. Besides, the adversary has a low total number of relayed messages in authentic encounters. To raise this amount and achieve well-behaving metrics, it has to set the forged number of sent messages higher than a lower bound as shown in Eq. 4.3. Therefore, the author identifies the appearance frequency and the number of sent messages of forged encounter records as the potential metrics to distinguish them from authentic records. The author will investigate those two metrics and show that colluding attackers cannot hide the abnormality of both metrics at the same time.

Figure 4.3: Encounter records frequency between node pairs.

To study the metrics, the author runs a simulation which involves 15 mobile nodes whose identifiers are numbered from 1-15. The first 6 nodes are chosen as attackers and they pair to collude (following the collusion model described earlier): (node 1-2), (node 3-4), (node 5-6). Fig. 4.3 plots the global view of encounter frequencies of all node pairs in the color matrix. The darker square means a higher value of encounter

Figure 4.4: Average number of sent messages between node pairs.

frequency for the corresponding pair. Fig. 4.3(a) shows the case where fake ERs forged by colluding pairs are excluded from computing the ER frequency. In general, it is hard to distinguish malicious pairs from normal ones. In Fig. 4.3(b), however, malicious pairs can be easily identified when fake ERs are included in computing the ER frequency. A similar trend can be found in Fig. 4.4 which plots the global view of average number of sent messages. The darker square means a higher value of sent messages number for the corresponding pair. Hence, encounter frequency and number of sent messages can be used as promising metrics for collusion detection.

It is noted that the collusion attackers cannot manipulate fake ERs and hide the abnormal pattern of both metrics at the same time. A malicious node can manipulate its forwarding ratio by creating one single fake ER with number of sent messages n_{send} and the risk is that n_{send} might be suspiciously high. Alternatively, the attacker can create more than one fake ER, say l fake ERs with the corresponding number of sent messages $n_1, \dots, n_l$. As both ways are equivalent to promoting the metrics, hence:

$$n_{send} = n_1 + \dots + n_l = \sum_{i=1}^l n_i$$

Using the latter option, the attacker can successfully reduce the number of sent messages per fake encounter (i.e. $n_1, \dots, n_l < n_{send}$) to a non-suspicious range. However, it

Figure 4.5: FXS metric of forged ER vs authentic ER.

has to raise the number of fake encounters from 1 to l instead.

Based on this observation, the author suggests a combined metric that accounts for both the number of sent messages and the encounter frequency as follows. Given an encounter window ERs of node j , for each node k recorded as encountering j in ERs , denote $freq_j^k$ as the number of encounters between j and k appearing in ERs , $send_j^k$ as the total number of messages sent from j to k over the window. The metric FXS is defined as

$$FXS_j^k = freq_j^k \times send_j^k$$

If j and k are colluders, FXS_j^k reflects the abnormality of fake encounters between them; thus FXS_j^k should be abnormally high. This is demonstrated in Fig. 4.5 which shows the histogram distribution of FXS in forged encounters and authentic encounters. As seen in the figure, most authentic encounters have FXS as low as 1000 while forged encounters normally have FXS as high as 2000 or more. The metric FXS of forged encounters is often much higher and can be differentiated from that of authentic encounters using threshold.

4.3.3 Detection of collusion

The author makes use of the above analysis to design the collusion detection scheme. When node i encounters node j , node i checks if node j has launched the collusion message-dropping attack in a 2-phase process: suspicion and confirmation. The first phase produces a list of suspicious colluders based on the FXS metric. The second phase is required to avoid false accusation of the suspected node who has high FXS metric because it actually has many encounters and exchanges many messages with node j .

Suspicion. In the first step, node i uses the ER history window provided by node j to infer the list of nodes that j has met in the window and the corresponding FXS metric. If the metric FXS_j^k exceeds a defined threshold Th_{FXS} , node k will be added to the suspicious list as it possibly colludes with j . Once suspected, step 2 is triggered to start the verification process.

Confirmation. In this step, the ERs of node j along with each node k in the suspicious list will be filtered from the current ERs window. Then the evaluation process described in Section 4.1.2 is implemented on the newly obtained ERs window, i.e. the suspiciously fake records will not be considered in the computation. This filtered window will have its relaying ratio RR' and own forwarding ratio SFR' calculated and checked against the thresholds Th_{RR} and Th_{SFR} . If the thresholds are violated, the same penalty is meted out to node j and to the corresponding node in the suspicious list.

Fig. 4.6 shows how the filtered RR and SFR are distributed for malicious nodes and normal nodes in the suspicious list. As seen in the figure, malicious nodes have lower RR and higher SFR than normal nodes though all of them get suspected due to abnormal FXS metric. It demonstrates that the filtered metrics can be well exploited

Algorithm 2 Algorithm of node i evaluating node j

Require: current ER window of node $j = ERW$
Require: $dropping = \text{false}$, $collusion = \text{false}$
Require: parameters Th_{RR} , Th_{SFR} , Th_{FXS} , γ , ρ , λ
 {individual attacker detection}

```

1: Calculate the metrics  $RR$  and  $SFR$  from  $ERW$ 
2: if ( $RR < Th_{RR}$ ) then
3:    $TR_j^i = TR_j^i - \gamma$ 
4:    $dropping = \text{true}$ 
5: end if
6: if ( $SFR > Th_{SFR}$ ) then
7:    $TR_j^i = TR_j^i - \rho$ 
8:    $dropping = \text{true}$ 
9: end if
  {cooperating attacker detection}
10:  $p = c_1$ 
11: for each node  $k$  that  $j$  encounter in  $ERW$  do
12:    $freq_k$  = number of times  $k$  and  $j$  encounters over  $ERW$ 
13:    $send_k$  = number of messages  $j$  sends to  $k$  over  $ERW$ 
14:    $FXS_k = freq_k \times send_k$ 
15:   if ( $FXS_k > Th_{FXS}$ ) then
16:      $ERW' = ERW$  excluding encounter records with  $k$ 
17:     Calculate the metrics  $RR'$  and  $SFR'$  from  $ERW'$ 
18:     if ( $RR' < Th_{RR}$ ) then
19:        $TR_j^i = TR_j^i - \gamma$ 
20:        $TR_k^i = TR_k^i - \gamma$ 
21:        $collusion = \text{true}$ 
22:     end if
23:     if ( $SFR' > Th_{SFR}$ ) then
24:        $TR_j^i = TR_j^i - \rho$ 
25:        $TR_k^i = TR_k^i - \rho$ 
26:        $collusion = \text{true}$ 
27:     end if
28:   end if
29: end for
30: if ( $dropping == \text{false}$ ) and ( $collusion == \text{false}$ ) then
31:    $TR_j^i = TR_j^i + \lambda$ 
32: end if

```

to differentiate malicious and normal nodes.

The author summarizes how the collusion detection is integrated with the individual

Figure 4.6: Filtered metrics of nodes in suspicious list.

detection in Algorithm 2.

4.3.4 Special case: collusion behavior-switching attack

To reduce the false positive rate, the author incorporates the reputation approach into the proposed detection scheme. Given a node n encountering a node k , if n judges k as well-behaving, n rewards k by increasing its reputation. Otherwise, n punishes k by reducing its reputation instead of immediately blacklisting k . Only if n could observe the repeated misbehavior of k over time which leads to very low reputation of k , then n will blacklist k .

However, malicious nodes could also make use of this feature to bypass the detection

as follows. Suppose a normal node takes T time units to observe and blacklist an adversary. Colluding adversaries first attack for T time units before being blacklisted, then behave well for T_2 time units to regain their reputation before continuing to attack. This misbehavior is called the collusion behavior-switching attack.

It is noted that this behavior-switching attack is theoretically feasible but hard to be implemented. The detection time T is different for different nodes as it depends on their respective encounters with the malicious nodes. No doubt, the malicious nodes can estimate T but they cannot ensure the accuracy of T . Hence the malicious nodes are still liable to be detected by some normal nodes.

To further suppress this attack, the author designs the scheme such that reputation is hard to build but easy to lose. The reward is set smaller than punishment so that adversaries will have their reputations dropped more quickly than increased. Meanwhile, behavior-switching attackers need to gain back the amount of reputation as much as the amount they lose. As a result, they have to behave well (for T_2 time units) longer than the attack period (for at most T time units), i.e. $T_2 > T$. Thus the attack frequency is mitigated. In Section 4.4.5, the author present simulation results to show the mitigation effect of the proposed scheme against this attack.

4.4 Performance Evaluation

4.4.1 Simulation setup

The simulation is implemented in ONE simulator [29]. The simulated network contains 40 nodes which can communicate in ad-hoc mode using Wifi in a transmission range of 100 m. They travel over an area of 4500 m $\times$ 3400 m, at speeds of 10 km/h to 50 km/h, using Shortest Path Map Based Movement Model which is available in ONE to

simulate the movement of vehicles on the streets. The simulation time is 43200 s (12 hours). Messages are generated at the rate of one per 25-30 s. The message size is in the range of 50 kB to 1 MB. This setting is applied to all the following experiments. For each experiment, the simulation runs for 10 times with random seeds and the average of the measured metrics are recorded and presented.

4.4.2 Evaluating the detection performance of SDBG

The detection performance of SDBG is assessed in two aspects: detecting ER manipulation and detecting collusion dropping. The following metrics for evaluation are used:

- Detection accuracy: percentage of malicious nodes that can be detected by normal nodes
- Detection delay: the time taken for the misbehavior to be detected.
- Detection false positive rate: percentage of normal nodes that are mistakenly judged as malicious by other normal nodes.

Detection of ER manipulation misbehavior

The total number of attackers is set to 12. Attackers hold back unfavorable ERs to hide their dropping behaviors, as described in Section 4.2.1. The author only considers this manipulation strategy because it is the most challenging to detect among three strategies.

ER manipulation ratio (EMR) is defined as the ratio of the number of manipulated ERs over all ERs for a malicious node. EMR of each adversary is varied from 0.05 to 0.4.

Figure 4.7: Detection delay of ER manipulation.

Table 4.2 shows the percentage of the normal nodes that can detect malicious nodes. It can be observed that most of the malicious nodes can be accurately detected by all the normal nodes. Fig. 4.7 illustrates 3 measures of detection delay: (i) min delay is the time taken for all malicious nodes to be detected at least once by any normal node; (ii) average delay is the mean delay over all normal nodes that can detect the misbehavior; (iii) max delay is the time that all malicious nodes have been detected by all the normal nodes. The detection delay is always highest at the manipulation percentage of 0.05 and lowest at the manipulation percentage of 0.4. When the manipulation percentage increases, the detection delay is reduced and detection rate is increased to 100%. This means the more frequently the attackers manipulate the encounter records, the faster and more possibly they can be discovered.

Table 4.2: SDBG's detection rate of ER manipulation.

EMR	0.05	0.1	0.2	0.3	0.4
Detection rate	0.98	0.99	1.0	1.0	1.0

Detection of collusion packet dropping misbehavior

The detection of collusion dropping is evaluated under (1) varying attacker setting (number of attackers, collusion setting, dropping probability) and (2) varying SDBG

Figure 4.8: SDBG's detection accuracy under varying attack settings.

setting (Th_{RR} , Th_{SFR} and Th_{FXS}).

Varying attacker setting. The total number of malicious nodes is set to 6 and 12, in which they form collusions of 2, 4 and 6 members. For example, if 12 attackers form collusions of 4 members, this case is represented as 4x3, i.e. there are 3 groups of attackers and each group consists of 4 colluding nodes. The dropping probability is varied from 0.4 to 1. The routing protocols used are Prophet [9], Maxprop [10] and Spray and Wait [11]. The thresholds Th_{RR} , Th_{SFR} and Th_{FXS} are fixed for each routing protocol.

Fig. 4.8 presents the detection accuracy of SDBG with varying routing protocols and attackers settings. From the figure, the author has the following findings: (a) for all routing protocols, SDBG achieves more than 70% accuracy, indicating good overall

Figure 4.9: SDBG's false positive rate under varying attack settings.

accuracy performance of SDBG; (b) when the number of colluders per group increases, the detection rate drops. Having more colluders helps a malicious node to distribute its fake ERs and hence lower the fake encounter frequency with each colluder; (c) when the dropping probability decreases, the detection accuracy also drops. Less suspicion is aroused if a malicious node reduces its dropping behavior.

The false positive rate of SDBG under varied contexts is plotted in Fig. 4.9. It is found that for all routing protocol, SDBG achieves a false positive rate lower than 1.6%. This shows that the two-phase scheme works efficiently in reducing the false positive.

The minimum detection delay under the various collusion settings are plotted in Fig. 4.10. It is observed that (a) for all routing protocols, the delay decreases when

Figure 4.10: SDBG's detection time under varying attack settings.

the dropping probability increases. Specifically, greyhole attackers takes a longer time to be detected than blackhole attackers. This is because the less probability that an attacker drops the messages, the less frequently it will violate the metrics, leading to longer time for its reputation to drop below the malicious threshold; (b) when the number of colluders per group increases, the delay also increases since it becomes more challenging to figure out a bigger group of colluders.

Varying thresholds setting. The attacker settings are fixed as follows: 12 attackers are divided into 2 groups of 6 colluders, the packet dropping probability is set to 0.5, the routing protocol is chosen to be Prophet. Only one of the three thresholds (Th_{SFR} , Th_{RR} , Th_{FXS}) in SDBG is varied at a time while the other two are fixed.

The results are shown in Figs. 4.11 and 4.12. As seen in Figs. 4.11(a) and 4.12(a),

Figure 4.11: Detection accuracy and false positive in Prophet under varying thresholds.

when Th_{RR} is increased, the detection accuracy is increased from 0.92 to 0.95 and the max detection time is reduced from 22000 s to 20500 s; but correspondingly, the false positive rate is increased from 1% to 3%. This is because malicious nodes tend to have lower RR than normal nodes. If Th_{RR} is raised, there is more chance that malicious nodes violate the thresholds and get detected earlier but at the same time more normal nodes are likely to be misjudged.

Similarly, the trend of simultaneous decreased detection accuracy, increased detection time and decreased false positive is observed as Th_{SFR} and Th_{FXS} increase. This demonstrates the trade-off among the three different performance metrics when the thresholds are varied.

Figure 4.12: Detection time of Prophet under varying threshold setting.

4.4.3 Comparing with other schemes

SDBG is compared with two state-of-the-art schemes, MDS [52] and Li's scheme [53] in the overall network performance when they are used to detect and prevent message dropping attackers. The following metrics are used for comparison:

- Number of wasted transmissions: number of messages that malicious nodes have received from normal nodes and then dropped intentionally.
- Delivery ratio: percentage of messages delivered to destinations out of total generated messages

- Delivery delay: average time taken for a message to reach the destination

Similar to SDBG, MDS proposes the use of statistical metrics based on encounter history to judge nodes but it does not consider dealing with collusion attacks. In Li's scheme, a node is judged as dropping messages if its current message buffer does not include all the messages it receives and stores in its previous contact. In collusion attack, an adversary can forge buffer by creating fake previous contact in which it claims it did not store or receive the dropped messages. Li's scheme addresses this collusion behavior by requiring each node to keep multiple recent contact records and present them for judgement. In this way, a normal node can see a longer contact history of a malicious node which likely includes an authentic encounter, and thus observe the messages it actually receives.

These schemes are evaluated under different scenarios where individual attacks or collusion attacks are launched. The number of attackers are set to 12. In individual attack, malicious nodes attack separately and cannot forge encounter records. In collusion attack, 12 adversaries are divided into 3 groups, each of which contains 4 colluders. The dropping probability is varied from 0.4 to 1. The routing protocols are Prophet and Maxprop.

Countering individual attack

Fig. 4.13 shows the overall network performance when using different schemes to detect individual attack. As seen in Fig. 4.13(a), MDS incurs at most 33000 and 9000 wasted transmissions for Prophet and Maxprop respectively. SDBG, on the other hand, incurs at most 31000 and 5000 wasted transmissions for Prophet and Maxprop respectively. Thus, SDBG always outperforms MDS in both routing protocols at any dropping probability. It shows that the suggested forwarding ratio metrics of SDBG

Figure 4.13: SDBG vs other schemes for individual attack.

can work better than those of MDS.

It can be observed that in MDS and SDBG, the plot of number of wasted transmissions has a valley point at the dropping probability of 0.8. At low dropping probability, the percentage of dropped messages is low but the detection accuracy is also low, providing adversaries more time to continue their attack. When the dropping probability is increased, the number of dropped messages is higher but the adversaries are detected faster and more accurately.

As seen in Fig. 4.13(b), MDS and SDBG are comparable in keeping the average delivery latency of at most 670 s for Prophet and at most 600 s for Maxprop. The delivery latency in Li's scheme is at least about 1200 s, which is about 2 times higher than that of MDS/SDBG. Based on the results in Figs. 4.13(a) and 4.13(b), Li's scheme

can reduce the number of wasted transmission more significantly than MDS/SDBG but the trade-off is that its delivery latency is much longer. Li's scheme cannot differentiate the dropping behavior of malicious nodes from normal nodes who drop with valid reasons. As a result, Li's scheme produces high false positive rate, making nodes miss many transfer opportunities with good relays and thus adds more delay to the message delivery time.

As seen in Fig. 4.13(c), the delivery ratio of MDS and SDBG are not significantly different, which is at least 0.72 for Prophet and at least 0.99 for Maxprop. Li's scheme can sustain the delivery ratio of about 0.96 for both Maxprop and Prophet. MDS and SDBG can achieve higher delivery ratio in Maxprop but lower delivery ratio in Prophet than Li's scheme. Compared to Maxprop, Prophet floods more messages for relay per encounter, leading to inefficient use of buffer storage. In Prophet, Li's scheme reduces a lot of relays due to its high misjudge rate, making the use of buffer more efficient and achieving a higher delivery ratio.

Countering collusion attack

Fig. 4.14 plots the network performance in the context of collusion attack. As seen in Fig. 4.14(a), the number of wasted transmissions of MDS is in the range of [90000-180000] in Prophet and [13000-30000] in Maxprop. As MDS cannot detect the collusion attack at all, attackers are not prevented and the number of wasted transmissions thus increases linearly with the dropping probability. Whereas in SDBG, the amount of wasted transmissions can be kept at 18000 and 8000 in Prophet and Maxprop respectively. As SDBG can detect colluders accurately, the number of wasted transmissions is kept stable with varying dropping probabilities. SDBG can reduce the wasted transmissions by at least 1.5 times compared to MDS and works especially efficiently in the presence of high dropping-probability attackers.

Figure 4.14: SDBG vs other schemes for collusion attack.

As seen in Fig. 4.14(b), the average delivery latency of both MDS/SDBG is at most 630 s and 500 s in Prophet and Maxprop. As seen in Fig. 4.14(c), the delivery ratio of MDS is lower than 0.7 in Prophet and about 0.99 in Maxprop. Whereas, the delivery ratio of SDBG is above 0.7 in Prophet and about 0.99 in Maxprop. It is observed that SDBG incurs a larger or equal delivery ratio and shorter delay compared to MDS but the difference is not so significant. Prophet and Maxprop are both replication-based routing protocols which allow a packet to be replicated and transmitted in multiple paths. Even if an attacker drops a packet in one path, the packet still has a chance to reach the destination via other paths. As a result, though MDS cannot detect and exclude collusion dropping attackers, it still achieves a comparable delivery ratio with SDBG.

In Li's scheme, the number of wasted transmissions is the lowest and the delay latency is the worst among the three schemes, which is also attributed to the high misjudgement rate. Although Li's scheme can deal with colluding adversaries, the high false positive rate results in the under utilisation of network connectivities. In contrast with Li's scheme, SDBG can detect attackers with high accuracy and low false positive. Most normal nodes are not excluded from the network by mistake so that the contact opportunities are exploited more efficiently and the average delivery delay is lower than Li's scheme. The number of wasted transmissions in SDBG is higher than Li's scheme within the simulation time because it needs more time to accurately judge the misbehavior. However, in the long term, SDBG can reduce the number of wasted transmissions as efficiently as Li's scheme because the malicious nodes are ultimately blacklisted correctly.

4.4.4 Storage overhead

There are two types of storage overhead: temporary and permanent. The temporary overhead is the encounter histories of other nodes that are stored temporarily and deleted immediately after the evaluation. The permanent overhead for each node includes its own ER history and meeting list (ML). The author only considers the permanent storage overhead. As each node needs to keep w most recent records in its history, the ER history overhead is:

$$Overhead_{ER} = w \times size_{ER} \quad (4.4)$$

Fig. 4.15(a) shows an overview of the fields in the ER and their corresponding sizes in bytes. The number of message records in sending list SL /receiving list RL is estimated as the average number of sent/received messages per ER. The author

runs a simulation and obtains the average number of records as 20 for Prophet routing protocol. As a result, the average size of ER is $4 \times 5 + 20 \times 12 + 20 \times 12 + 40 \times 2 = 580\text{B}$. As w is set to 100 by default, the total storage overhead of ER history is $100 \times 580\text{B} = 58\text{kB}$.

Each node also stores the meeting summary (MS) of all its neighbors in ML, thus the ML overhead is:

$$\text{Overhead}_{ML} = n \times \text{size}_{MS}$$

where n is the number of neighbors. For the simulation setting, n is 39. As seen in Fig. 4.15(b), the MS size is 16B. The total ML overhead is $39 \times 16 = 624\text{B}$, which can be ignored compared to the ERs overhead (58kB). Therefore, the author can consider the total storage overhead as being equivalent to the ER overhead which is 58kB. This total amount of overhead is affordable for mobile devices such as mobile phones and negligible for notebooks.

(a) Encounter Record (ER).

(b) Meeting List (ML).

Figure 4.15: SDBG's data overhead size.

Choosing window size

According to Eq. 4.4, ER history overhead increases linearly with the window size w . Thus it is necessary to reduce w for nodes with very limited storage to reduce their overheads. However, there is a trade-off between the SDBG's detection performance and the window size. The author plots the detection accuracy and false positive over various window size in Fig. 4.16. The simulation setting is 3 groups of attackers with 4 colluders in each group and the dropping probability is 0.6. Prophet routing protocol is used.

Figure 4.16: SDBG detection performance under varying window size.

Fig. 4.16 shows that the larger the window, the higher is the detection accuracy and the lower is the false positive rate which SDBG can achieve. When w increases, the detection accuracy experiences a sharp increase at w of 30, then stabilizes thereafter. A similar trend is observed for the false positive rate. This fact allows nodes to choose smaller w that can reduce the storage overhead significantly without sacrificing the SDBG performance a lot. For example, devices with very limited storage can choose a medium window size such as 30. This reduces the storage overhead by 1/3 compared to using window size of 100 but only degrades the detection accuracy and false positive a

bit compared to the best SDBG performance. Nodes can choose an appropriate window size that fits their storage availability and their requirements on SDBG performance.

4.4.5 Mitigating collusion behavior-switching attack

The author runs simulations to demonstrate the effect of mitigating the collusion behavior-switching attack (described in Section 4.3.4) using SDBG. The simulation settings are as follows. There are 3 groups of attackers with 2 colluders in each group. The dropping probability is 1.0. The window size is 100. The routing protocol is Prophet. The number of runs is 10.

First, the malicious nodes are set to attack for a period of T and to behave well for the rest of the simulation time. T is chosen based on the results of detection time in Fig. 4.10 (a) for the settings above. The average detection time over 10 runs is 6829 s. For each run, the author measures the minimum detection time which is the shortest time to detect at least one malicious node. After 10 runs, the author obtain the minimum detection time as 639 s and the average minimum detection time as 2230 s. T is varied from the minimum detection time to the average detection time.

Fig. 4.17(a) shows the average detection accuracy of SDBG over attack period T . It is observed that even when T is set to the minimum detection time, the detection accuracy is low (2%) but still larger than 0. When T is set to the average detection time, the detection accuracy can reach 100%. This is because even when the attackers stop attacking, parts of their misbehaviors are still reflected in their ER histories. Thus, nodes that encounter the attackers early after the attack period might detect the malicious nodes.

After the attack period T , the reputations of the malicious nodes drop and they need to behave well for T_2 to recover their reputations to the initial value TR_{init} . Fig.

4.17(b) plots the average and maximum recovery time over T . It is noted that when T is set to the average detection time (6829 s), the recovery time is 0 as all attackers are detected and cannot recover their initial reputations at all. Besides, the average recovery time is 3 to 6 times larger than the attack period T . This demonstrates that the proposed reward/punishment scheme can effectively mitigate the collusion behavior-switching attack.

(a) Detection accuracy.

(b) Recovery time.

Figure 4.17: SDBG under varying attack period T .

Next, the author sets the malicious nodes to alternate between bad and good behaviors for T and T_2 respectively. For each value of T , T_2 is set to the average and maximum recovery time obtained in Fig. 4.17(b). Fig. 4.18 shows the detection accuracy over T under such setting. When T_2 is set to the average recovery time, the detection accuracy is larger than 60%. Even when T_2 is set to the maximum recovery

time, it is still possible for the attackers to be detected. Hence, to avoid being detected, attackers have to decrease T and increase T_2 , which in turn leads to insufficient attacking periods. Moreover, the result also demonstrates that it is hard for attackers to compute T and T_2 accurately to conceal their misbehaviors.

Figure 4.18: SDBG's detection accuracy in behavior-switching attack.

4.5 Conclusion

SDBG is proposed to protect DTN against both individual and collusion black-hole/greyhole attack in DTN. Simulation results show that SDBG works with a wide range of packet-dropping probability, different routing protocols and varying number of colluders. SDBG outperforms the state-of-the-art works in both detection accuracy and false positive rate. SDBG is also robust against the behavior-switching attack. Moreover, the storage overhead incurred by SDBG is affordable and adjustable to fit the node's requirements and constraints.

Chapter 5

Detecting Flooding Attack in DTN

In this chapter, the author focuses on the flooding attack in DTN. The author introduces burst-limit policy to enable nodes to generate a short outburst of messages legitimately and proposes FDER (Flooding Detection based on Encounter Record) to detect the adversaries that violate this policy. Additionally, the author designs a forwarding and queuing policy as the compensation of burst-limit policy to ensure the delivery performance fairness among nodes with different traffic demands. Simulations show that the proposal can counter against flooding attack and enable flexible and fair transmissions simultaneously.

5.1 Overview

5.1.1 Background and motivation

Due to intermittent connectivity, it is hard to establish end-to-end path from the source to the destination of a message. Instead, DTN's routing strategy is "store-and-forward": nodes first stores the message and then forwards it to the appropriate next hop or the destination opportunistically. To increase the delivery success, most routing protocols in DTN allow messages to be replicated and disseminated in different paths. In some routing protocols such as Epidemic [8], Prophet [9], Maxprop [10], there is no limit on the replica number for each message. Specifically, after a node relays a message to another node, it still keeps that message to forward in later contact opportunities. Other routing schemes such as Spray and Wait [11] aim at limiting the total number of replicas for each distinct message in the network. Thus, the source and relay nodes of the message can forward it for a limited number of times which is defined by the routing protocol. The former routing strategy is referred to as propagation routing and the latter is referred to as quota-based routing.

Utilizing the multi-copy feature of DTN routing protocols, malicious nodes may launch the flooding attack as follows. Malicious nodes generate and send too many unnecessary messages, which in turn would be replicated into huge amount of bogus data over the network. Moreover, in quota-based routing, malicious nodes may replicate each of their messages for a number of times higher than the defined threshold. The former attack strategy is referred to as message flooding and the latter replica flooding. Both flooding strategies spawn invalid data that contend for network resources with legitimate data and ultimately reduce the delivery rate of normal nodes' messages. This motivates the author to design a scheme to protect DTN against flooding attack.

5.1.2 Defense lines against flooding attack

In this section, the author states the defense mechanisms used to mitigate the flooding attack described above. The first defense line is authentication. Legitimate nodes own valid keys and sign their own messages so that other nodes can authenticate the messages' source nodes. Nodes only keep and forward messages generated by internal nodes. Thus authentication deters external flooding attackers that inject unauthorized data into the network. However, authentication cannot detect internal adversaries that deliberately flood messages or replicas.

The second defense line is rate limit introduced in [66]. Suppose time is divided into equal fixed slots of duration T , starting from 0, T , $2T$ etc. Each node has a rate limit M which is the maximum number of new messages that it is allowed to generate and send per time interval T . Rate limit mitigates the message flooding attack but fixed rate constrains nodes from sending messages on demand. Flexible rate is possible [66] by allowing nodes to request the authority for new rate-limit permissions when their sending demands change. However, this solution is infeasible if the connectivity to the authority is limited or costly especially if the nodes' sending demands are unpredictable.

Thus the author introduces burst limit as an alternative to rate limit that flexibly accommodates the nodes' short-period burst traffic. Specifically, nodes are allowed to exceed the rate limit by at most K messages per slot and have at most Q consecutive burst slots. An interval, in which a node sends more than M messages but less or equal than $M + K$ messages, is called a valid burst slot. In this way, the excess messages in valid burst slot still get some chance of being delivered if the network resources are sufficient. Nodes that violate these rules are deemed as attackers.

The third defense line is replica limit which is the maximum number of copies for each message that a node is allowed to forward in the network. The replica limit

is normally defined by the routing protocol. For example, Spray and Wait [11] with replica limit L only allows the source node of a message to forward that message for at most L times. This aims at mitigating the replica flooding attack.

5.1.3 System model

Network model

The network consists of mobile nodes communicating in ad-hoc mode using wireless technology such as WiFi. Each node owns a unique identifier. The network is loosely time synchronized such that if the time is divided into equal-length time slots, any two nodes are in the same time slot at any time.

The author assumes authentication is enabled by identity-based cryptography (IBC) which involves a key generator center to generate a valid private key for each node from the node's identifier. Due to no requirement of cumbersome infrastructures and light overhead, IBC is suitable for mobile ad-hoc network like DTN [68].

The author adopts the burst limit and replica limit mentioned above as the approach to mitigate internal flooding attack in DTN.

Attack model

Attackers can flood messages or replicas or both. During flooding, an attacker either exceeds its rate-limit M by more than K messages per interval or continuously exceeds its rate-limit M for more than consecutive Q intervals. When flooding replicas, an attacker exceeds the replica limit L which is defined by the routing protocol. Fig. 5.1 summarizes the factors involved in the different attacks. It is noted that, since there is no replica limit in propagation routing protocols, replica flooding attack is

only considered in networks using quota-based routing protocols.

Figure 5.1: Factors of flooding attack.

5.1.4 Basic ideas

Adopting rate/burst and replica limit to mitigate flooding attack involves evaluating if a node has generated more messages or forwarded more replicas than the thresholds. However, monitoring how many messages and replicas that nodes forward is non-trivial in DTN since data are routed opportunistically and no single node can keep track of all the sending activities of other nodes.

To resolve this issue, the author introduces the message header and encounter record (ER) as the tools for nodes to monitor how many messages and replicas other nodes have forwarded in a distributed manner. Specifically, the header of a message is created by the source node to specify the order in each time slot in which the message is sent for the first time, i.e. message count. The ER records the messages that a node has sent and received during an encounter with another node by specifying the message headers. For each record of sent message in ER, the sending node also claims the number of times it has replicated that message, i.e. replica count. Each node keeps its most recent ERs and presents them to other nodes it encounters so that these nodes can evaluate its transmission behavior. An attacker that floods messages or replicas

will have its message counts or replica counts higher than the given limits. Thus by looking at the adversary's ERs, other nodes can detect the flooding attack.

5.2 Misbehavior detection scheme

In Sections 5.2.1 and 5.2.2, the author presents the construction of message header and ER and how they are used to detect message and replica flooding attacks. Since message headers and ERs reflect the nodes' transmission history, they enable monitoring if nodes' transmissions exceed the given limits. The adversaries may hide their flooding attacks by manipulating their ERs or the actual fields in the ERs. The author describes these misbehaviors in detail and proposes the corresponding detection procedures in Sections 5.2.3, 5.2.4 and 5.2.5.

5.2.1 Message model

The time is divided into fixed time slots of length T , starting from 0, T , $2T$, etc. When a node has something new to send, it first generates the message content. When it has a chance to send the message for the first time, it creates the message header to record the source and destination nodes, the current time and the message count. The message count reflects the order in which this message is sent out for the first time in this time slot. Finally, the source node signs off the message content and it is concatenated with the message header. The signature ensures the message is created by a legitimate node and protects the content and header from being modified by intermediate nodes.

For a new message m , the source node creates the message header H_m which is formatted as follows:

$$H_m = \langle S, D, ts, c \rangle$$

$$m^* = \langle m, H_m, sig \rangle$$

where S and D are the respective identifiers of m 's source and destination nodes, ts is the current time slot when S first sends m and c is the message count of m (i.e. m is the c^{th} new message that S has created and sent out in ts), sig is the source node's signature over the content and header, m^* is the concatenation of message m , header H_m and signature sig .

The message header and signature are always forwarded together with the message content to later hops (i.e. m^* is replicated in relay nodes). When an intermediate node receives m^* , it first verifies the signature and checks if c exceeds $M + K$. If the signature is invalid, it blacklists the previous hop since it forwards a corrupted message. If the signature is valid but c is larger than $M + K$, it blacklists the source node of m^* and alerts other nodes about the blacklisted node. Otherwise, it keeps m^* in its local storage.

Alert of the adversaries

Once an adversary is detected, the alert procedure is implemented as follows. The node that has authenticated evidences to prove the misbehavior of the adversary will add it to the blacklist and keep the evidences. From then on, when it encounters other nodes, it forwards the blacklist to the peers. If the peers have not blacklisted any listed adversary, the peers will request the node to send the corresponding evidences. The peers first verify the evidences; if they are valid, they blacklist the adversary and help in further alerting the network. Since the evidences are authenticated, badmouthing attack, in which a node accuses a normal node as an adversary, is impossible.

Setup	<ol style="list-style-type: none"> 1. Let G_1, G_2 be two cyclic groups of prime order q, P be the generator of G_1, 2. Let $e : G_1 \times G_1 \rightarrow G_2$ be a bilinear pairing which satisfies the following 3 conditions: <ol style="list-style-type: none"> (a) <i>bilinear</i>: $\forall x, y \in G_1, a, b \in \mathbb{Z}_q, e(a^x, b^y) = e(x, y)^{ab}$, (b) <i>non-degenerate</i>: $e(P, P) \neq 1$ (c) <i>computable</i>: $\exists$ efficient algorithm to compute $e(x, y) \forall x, y \in G_1$ 3. Pick a random $s \in \mathbb{Z}_q$ and compute $P_{pub} = P^s$ 4. Choose two hash functions $H_1 : \{0, 1\}^* \rightarrow G_1$ and $H_2 : \{0, 1\}^* \times G_1 \rightarrow \mathbb{Z}_q$ 5. The system parameter is (P, P_{pub}, H_1, H_2). The master key is s.
Extract	Given an identity ID , compute $Q_{ID} = H_1(ID)$ and $D_{ID} = Q_{ID}^s$. The private key is D_{ID} .
Signing	Given a private key D_{ID} and a message m , pick a random $r \in \mathbb{Z}_q$ and compute $U = Q_{ID}^r$, $h = H_2(m, U)$, $V = D_{ID}^{h+r}$. The signature is $\sigma = (U, V)$.
Verification	Given the signature is $\sigma = (U, V)$ of message m for an identity ID , compute $h = H_2(m, U)$. Accept the signature if $e(P, V) = e(P_{pub}, U \cdot Q_{ID}^h)$.
Batch verification	Given η signatures $(ID_1, m_1, U_1, h_1, V_1) \dots (ID_\eta, m_\eta, U_\eta, V_\eta)$, compute $Q_i = H_1(ID_i)$ and $h_i = H_2(m_i, U_i)$. Accept the signatures if $e(P, \prod_{i=1}^\eta V_i) = e(P_{pub}, \prod_{i=1}^\eta U_i \cdot Q_i^{h_i})$.

Figure 5.2: Identity-based cryptography signatures and batch verification

Batch authentication

Each node is required to verify the signatures of all the received messages, which will incur high overhead for a large number of messages. To reduce the authentication overhead, the author adopts the batch authentication technique that allows a node to verify multiple messages which are signed by multiple users in a fixed number of operations which is independent of the number of messages.

In particular, the author uses batch verification scheme of IBC-based signatures [69] which is enabled by the bilinear feature of pairings. The algorithm details are summa-

rized in Fig. 5.2. The verification method requires several types of operations including pairing computation (i.e. $e(x, y)$), hashing (i.e. H_1, H_2), scalar multiplication, elliptic curve addition. Since pairing is much less efficient than other operations, the verification efficiency thus depends on the number of pairings. As seen in Fig. 5.2, if η messages are individually verified, the number of pairings is $2 \times \eta$. Meanwhile using the batch verification, η messages can be verified in 2 pairing operations only. Thus the verification cost is significantly reduced, especially when the number of messages η is high.

5.2.2 ER model

Generating ER

When two nodes encounter each other, they create an ER that records their identities, the ER sequence numbers assigned by them, the encounter timestamp and the list of exchanged messages. For each message that a node forwards in this encounter, the node records the message header information and the number of times it has replicated this message so far in the ER. Finally, both nodes sign off the ER content to prevent each side from modifying the record and store this authenticated ER in their local storage. Due to the limitation of storage, each node only keeps a window of W most recent ERs.

When a node i encounters another node j , node i generates ER_i :

$$ER_i = \langle ID_i, ID_j, sn_i, sn_j, t, SL_{ij}, RL_{ij} \rangle$$

$$SL_{ij} = \{MR_{ij}^m \mid i \text{ sends message } m \text{ to } j\}$$

$$RL_{ij} = \{MR_{ji}^m \mid j \text{ sends message } m \text{ to } i\}$$

$$ER_i^* = ER_i, sig_i, sig_j$$

where ID_i, ID_j are their respective identifications, sn_i, sn_j are the sequence numbers assigned to the ERs by nodes i and j respectively; t is the encounter time; sig_i, sig_j are their signatures over ER_i using their respective private keys. A node assigns its new ER with the sequence number of its latest ER incremented by 1. SL_{ij} and RL_{ij} denote the lists of message records MR that node i sends to and receives from node j . MR_{ij}^m , denoting the record of message m that node i sends to node j , comprises the fields shown as follows:

$$MR_{ij}^m = \langle H_m, r_{im} \rangle$$

where H_m denotes the message header of m as mentioned in Section 5.2.1. r_{im} indicates the replica count, i.e. this is the r_{im}^{th} time that the sender i (either as intermediate or source node of m) has forwarded m . It is noted that node i cannot modify H_m to badmouth the source node (for example, by claiming a very large MSG_m) since j can verify the message header which is protected by the source node's signature. Node i stores the authenticated ER_i^* in its local storage.

Detecting flooding attack with ER

When two nodes i and j encounter each other, they exchange their most recent ER windows to evaluate each other's sending behavior. Node i first verifies the signatures of the ERs submitted by node j . Batch authentication mentioned in Section 5.2.1 is used to accelerate the ER authentication. If node j 's ER window passes the verification, node i proceeds to check if j exceeds the burst limit as follows. From node j 's ER history, node i traverses through the lists of sent messages of node j and extracts the list SSL_j recording the times node j sends its own messages:

$$SSL_j = \{MR_{jx}^m \mid \text{the source node of } m \text{ is } j \text{ and } j \text{ sent message } m \text{ to some node } x \text{ in the given ERs of node } j \}$$

Based on SSL_j , node i can infer the maximum message count of node j over the past time slots. Node i checks if any maximum message count exceeds $M + K$ or if there exists $Q + 1$ consecutive slots in which all the maximum message counts exceed M . Furthermore, if quota-based routing protocol is used, node i checks each message record in SSL_j to see if any replica count exceeds L .

If node i detects that node j violates burst or replica limit, it will blacklist j and alert other nodes about the adversary j . Node j also performs the similar authentication and evaluation process on node i 's ER.

Attack model against ER

To avoid the flooding attack detection, an attacker may launch attacks against ER

1. manipulating ER: the adversary deletes unfavorable ER from its history
2. manipulating message count: the adversary declares wrong message counts in ER
3. manipulating replica count: the adversary declares wrong replica counts in ER

Each of these misbehavior is analyzed respectively in Sections 5.2.3, 5.2.4 and 5.2.5. It is assumed that there is no collusion attack in which the adversaries cooperate with one another to create fake ERs or incorrect ER information.

5.2.3 Detecting ER manipulation

Misbehavior

An attacker may delete unfavorable ERs (i.e. those revealing its misbehavior) from the ER history to obtain an ER window beneficial for itself to present to other nodes. For example as in Fig. 5.3, the adversary's actual ER windows are A , B , C , D with

the respective timestamps of 10, 20, 30, 40 and respective sequence number (assigned by attacker) of 1, 2, 3, 4. If it wants to delete the ER *C* with the sequence number 3, it can use one of the following three manipulation strategies:

1. *Skip ER sequence number.* The adversary deletes ER *C* and does not use the sequence number 3 in any other ER.
2. *Reuse ER sequence number.* The adversary deletes ER *C* and assigns the sequence number 3 to ER *E* that occurs at the later timestamp of 50.
3. *Hold back unfavorable ERs.* The adversary holds its sequence number at 3 to generate different ERs i.e. *C* and *D*. After achieving a favorable ER, it will increment the sequence number for later new ERs.

Figure 5.3: ER manipulation strategies.

Observation

According to the rule of creating ER in Section 5.2.2, a series of consecutive well-behaved ERs has sequential sequence numbers. Besides, ER with higher sequence

number has a bigger timestamp. However, adversaries using strategies 1 and 2 described above violate either rules. In the first strategy, the new ER history has the series of sequence numbers = 1, 2, 4 which is not sequential. In the second strategy, the new ER history has the series of time stamps = 10, 20, 50, 40 which is not monotonically increasing. When the manipulated ER window is presented to a neighbor node in later encounter, the misbehavior is easily detected.

When adversaries use strategy 3, the manipulated ERs history is A, B, D which abides by both sequence number and timestamp rules. Thus the neighbor cannot discover the manipulation at all if it only depends on the ER window presented by the malicious node. However, the misbehavior leads to inconsistency in that at least two different records C and D have been assigned with the same sequence number by the adversary.

Fig. 5.4 illustrates how to detect the adversary q that uses the same sequence number 3 to create ERs C and D with two different nodes i and j . If nodes i and j meet each other and exchange their ERs, they will directly discover that node q has manipulated its ERs. However, if node q selectively manipulates the ERs with two nodes i and j that rarely meet, the possibility that q gets detected is much lower. To deal with this strategy, there is a need of another node k who meets both i and j and stores their ERs C and D locally. Upon encountering i and receives its ERs, k notes and stores ER C locally. If later on, k also encounters j , it can check ER D of j against ER C and discover that q has assigned the sequence number 3 to different ERs.

This detection process requires node k to store several ERs of node i it encounters. To achieve efficient detection at low cost, the author proposes the sampling storage process as follows. Each node is required to maintain a global ER database (GERDB) in its local storage. When the node receives an ER history of a neighbor, it randomly chooses a few ERs of the neighbor and stores them in its GERDB. Since storing all

Figure 5.4: Detecting ER manipulation.

the received ERs is expensive, each node is only allowed to sample a small number α of the received ERs and limit the maximum number of ERs in GERDB to β . When GERDB's capacity is full and new ERs are added, the oldest timestamp ER is deleted to make room for the new one. As a result, GERDB provides each node with a partial global view of sequence number assignment over the network.

Detection

When node i encounters and receives the ER window of node j , node i first examines if node j 's ERs have their sequence numbers assigned by node j follow the sequential order and be consistent with the timestamp order. Next node i cross checks node j 's ERs against the ERs in its GERDB to inspect if any node has assigned the same sequence number to different ERs. If node i detects any inconsistencies due to any node, it will blacklist that node and alert other nodes.

5.2.4 Detecting message count manipulation

Misbehavior

Fig. 5.5 shows an example of an attacker that claims the wrong message count when it launches message flooding attack. The ER window size W is set to 4, the rate-limit M is set to 3 but the attacker has generated and sent out 4 distinct messages m_1 , m_2 , m_3 and m_4 in the current time slot 1. If it keeps the count authentic, the message header of m_4 reveals its violation of rate limit. To avoid being detected easily, the attacker can claim the message count with a smaller value (such as 1) than the real value, i.e. reusing message count.

Figure 5.5: Manipulating message count.

Observation

An adversary can use the same message count 1 for m_1 and m_4 to hide its flooding attack. However, when m_1 and m_4 are disseminated in the network, they will be deemed as the same message (despite their contents being different) since they have the same source node, message count and time slot. For existing routing protocols in DTN, upon encounter, nodes only exchange messages which they have not relayed

by comparing their lists of stored messages. Thus the intermediate nodes that already keep m_1 in their buffers will not accept m_4 and vice versa. The total number of replicas of m_1 and m_4 in the network is therefore restricted.

To increase the number of nodes to receive either m_1 or m_4 , the adversary q may assign m_4 a destination different from that of m_1 . However, this misbehavior leads to the inconsistency of destination nodes between two messages with the same source node, message count and timestamp. A node i could detect this misbehavior in two scenarios:

- Suppose node q sends m_1 and m_4 frequently such that the record of both messages are present in its ER window. When node q submits this ER window to an encountered node i , node i will observe the destination inconsistency.
- Suppose node i already stores m_1 . Later, node i may discover about m_4 when it encounters a node whose ER history involves the record of m_4 . Another way for node i to observe m_4 is when it encounters a node that already stores m_4 and they compare their lists of messages as part of the routing process. In either case, node i can detect the destination inconsistency.

Detection

When node i encounters node j and receives its ER window, node i checks if there exists any pair of message records in j 's ER history such that they have the same source node, same message count and equal sent time slot but different destination nodes. Node i also checks the message records in node j 's ERs against the headers of messages that node i stores to inspect the destination node inconsistency. Upon detecting any inconsistency between a pair of message records or headers, node i blacklists the source node of that message and alerts the network.

5.2.5 Detecting replica count manipulation

Misbehavior

When an attacker launches replica flooding attack and violates the replica limit, it may reuse existing replica count of the same message to hide its misbehavior as shown in Fig. 5.6 (a). The ER window size W is set to 4, the replica limit L is set to 3 but the attacker has forwarded the message m for 4 times. If it honestly declares the replica count, the fourth count is easily detected as being higher than the replica limit. Another choice for the attacker is to claim a replica count of a smaller value (such as 1) than the actual value, i.e. reusing replica count.

Figure 5.6: Manipulating replica count.

Observation

For a normal node that declares the correct replica count, the series of replica counts for each message in increasing order of sent timestamp is sequential. Note that the sent timestamp of a replica is equivalent to the creation timestamp of the related ER. However, when the adversary reuses replica count to hide its misbehavior, it violates the above observation. As shown in Fig. 5.6(a), if the adversary reuses the replica count 1 for message m in ER E , the ER window (B, C, D, E) reflects that the series of replica counts of m in the increasing order of sent time, which is 2, 3, 1, is non-sequential.

To prevent the replica count reuse strategy from being detected easily, the attacker may flood replicas at a later time such that the inconsistent replica counts are not present in the same ER window. For example, in Fig. 5.6(b), the adversary sends superfluous replicas of message m in ERs X, Y, Z when ERs A, B, C which record the previous replication of m are out of its ER window. When the adversary presents its ER window, its misbehavior cannot be detected directly. However, it can be detected if there exists a node that knows about both ERs A and X and reports the replica count inconsistency in these ERs.

Fig. 5.7 illustrates the detection process of the adversary q that claims the same replica count 1 when it replicates the message m to two different nodes i and j in the respective ERs A and X . Node q is detected if nodes i and j meet each other, exchange ERs and check the replica count records in its ERs against the peer's ERs. Node q is also detected if node k meets both i and j and stores their ERs A and X locally. Specifically, upon encountering i and receiving its ERs, k notes and stores ER A locally. If later on, k also encounters j , it can check ER X of j against ER A and discovers that q has reused the replica count. To implement this detection, the author makes use of GERDB mentioned above to enable node k to sample node i 's ERs and

opportunistically detect node q .

Figure 5.7: Detecting inconsistent assignment of replica count.

Detection

When node i encounters node j and receives its ER window, node i extracts the replica counts for each message m that node j has sent out and checks if this sequence is sequential. If the sequence is not sequential, node i blacklists node j and alerts the network. Node i also checks the replica count records in node j 's ERs against those in node i 's ER window and node i 's GERDB. If there exists any pair of different records that indicates the same node has sent out the same message and claimed the same replica count, that node is blacklisted and the network is alerted by node i .

5.2.6 How ER window mitigates flooding attack

This section explains the role of ER window in mitigating the flooding attack by showing the attacker's trade-off between its flooding capability and the rate of reusing replica and message counts.

Fig. 5.8(a) shows the trade-off for attackers in replica flooding. The window size W and replica-limit L are set as 4 and 3 respectively. It is observed the more replicas the

(a) Trade-off in flooding replicas.

(b) Trade-off in flooding messages.

Figure 5.8: Attacker's trade-off between the number of flooded replicas and the rate of reusing count.

adversary floods, the longer it needs to wait to reuse the replica and message count. If the attacker sends 3 replicas of message m in ERs A , B and C , it needs to wait until ER G (6 ERs from A) to reuse the replica count 1 for m . Meanwhile, if it sends only 1 replica in ER A , it can reduce the reuse waiting time to ER E (4 ERs from A) but this will also reduce its flooding efficiency.

Similarly, Fig. 5.8(b) shows the trade-off for attackers in message flooding. The window size W and rate-limit M are set as 4 and 3 respectively. Suppose the attacker uses the same message count for m_1 and m_2 but different destination nodes. If the attacker sends 3 replicas of message m_1 in the first three ERs, it needs to wait 6 ERs until reusing the message count 1 for m_2 . In contrast, if it sends only 1 replica, it can

reuse the message count earlier but misses the chance to disseminate more replicas of message m_1 .

From the example in Fig. 5.8, it can be generalized that to reuse a message or replica count, the attacker needs to wait for at least W ERs to pass. Thus setting a larger value for the window size W can help reduce the count reuse rate of attackers. Note that even if attackers can reuse the counts without being detected immediately, its misbehavior could still be detected opportunistically as presented in Sections 5.2.4 and 5.2.5. The author evaluates how window size affects the detection performance by simulation later in Section 5.4.5.

5.3 Fairness Forwarding and Queueing Policy

Although burst-limit policy provides nodes with more flexible transmission, it brings about the concern whether the nodes that generate frequent high traffic occupy more network resources than those that generate lower traffic. To compensate the effect of burst traffic and to mitigate the effect of flooding attack when it is not detected yet, the author designs a fairness policy to decide the priority of messages to be sent when a forwarding opportunity arises and of deleting them from the message queue. The main principle of the policy is that if a node is observed to send out many new messages recently, other nodes will give its messages less priority to be forwarded.

5.3.1 Updating the forwarding priority

The author proposes that each node maintains and updates a metric called the forwarding priority for other nodes in the network. By observing the ERs provided by neighbors and the receiving messages, a node learns the maximum message count per

time slot as declared by other nodes. The node maintains the maximum counts of other nodes in the most recent slots and uses them to update the forwarding priorities of other nodes. If the maximum count of a node is below the rate limit, its forwarding priority is raised and vice versa.

Denote f_j^i as the forwarding priority that node i gives to the messages generated by node j . The range of f_j^i is between 0 and 1. It is initiated with a default value $F \in [0, 1]$. Node i maintains the message summary (MS) to record the maximum message counts of node j for the most recent R slots:

$$MS_j^i = \langle ID_j, f_j^i, c(G), \dots, c(G - R + 1) \rangle$$

where ID_j denotes the identifier of node j , $c(g)$ is the maximum message counts of messages generated by node j in the slot g ($\forall g = G, G - 1 \dots G - R + 1$) that are known to i , G is the current time slot. Node i can learn the message counts of messages generated by node j when (1) node i meets another node that presents ERs involving the messages generated by node j , (2) node i receives messages generated by node j upon opportunistic encounter with other nodes. As ERs and the messages are authenticated with signatures, the message count value is ensured to be authentic. The update can only occur if the new maximum count is larger than the current maximum count for the same time slot. If node i has no information about node j message history in a slot g , the maximum count $c(g)$ is set to -1 . Note that $c(g)$ update can be piggybacked on the misbehavior detection process presented in Section 5.2, so little overhead is incurred.

When node i removes the oldest maximum message count $c(g)$ from MS_j^i , node i updates f_j^i as follows. If $c(g)$ is equal -1 , node i does nothing since it has no information about node j 's message history during this time slot. If $c(g)$ is equal or less than the rate limit M , node i increases the forwarding priority:

$$f_j^i = \max(1, f_j^i \times \delta_1)$$

where δ_1 is a constant > 1 . Otherwise, node i decreases the forwarding priority proportionately to its rate of sending new messages:

$$f_j^i = f_j^i \times (M \div c(g))_2^{\delta_2}$$

where δ_2 is a constant > 1 . Before $c(g)$ is deleted, node i check if node j has r ($\leq Q$) consecutive outburst time slots (i.e. maximum count exceeds the rate-limit M) starting from this slot g . If this is the case, node i further decreases f_j^i :

$$f_j^i = f_j^i \times (\delta_3)^r$$

where δ_3 is a constant < 1 .

Besides, the author considers fairness in the aspect of rewarding the effort of each node which helps to forward the messages of other nodes. When node i meets node j , node i can calculate SO_j and SI_j which are the respective number of messages that node j sends for other nodes and for itself during recent encounters. To compensate for the forwarding effort of node j , node i can increase f_j^i if the ratio of $SO_j \div SI_j$ is larger than a certain threshold:

$$f_j^i = \max(1, f_j^i \times \delta_4)$$

where δ_4 is a constant > 1 .

5.3.2 Fairness policy

Based on the calculated forwarding priority, the author designs the message forwarding and queuing policies as follows. When node i encounters node j , they exchange

messages with each other. Node i will forward the messages based on their source node and in the order of decreasing forwarding priority. When node i receives a message but its buffer runs out of space, it needs to delete one or several messages to make room for the new message. Node i will delete messages in the order of increasing forwarding priority of their source nodes.

It is noted that the maximum message counts recorded by node i may not be the actual maximum counts of node j in a certain time slot. Thus the local value of f_j^i maintained by node i might be higher than reality. However, with the opportunistic contact and exchange of ERs among nodes, it is highly possible for node i to obtain enough information to make an acceptable estimate of f_j^i . Even if node i overestimates f_j^i and gives high priority to node j 's messages, the forwarding of these messages could be limited in other nodes who have more information about j . Thus the superfluous messages of node j can be mitigated in the network.

The forwarding and queueing policies help to mitigate the effect of the malicious node's strategy of bursting the set threshold. Consider a malicious node that sends more than M but less than or equal to $M + K$ new messages per slot (i.e. valid burst slot) for consecutive Q slots, then sends less than M messages (i.e. rate-limit slot) for the next slot, and continues the pattern of alternating between burst and rate-limit traffic. In this way, the malicious node can bypass the detection of message flooding. However, since the burst is too frequent, its forwarding priority is often being reduced more than increased, leading to a very low forwarding priority. As a result, the messages of the malicious nodes can be efficiently limited.

5.3.3 Integrating fair policy with existing routing protocols

Existing routing protocols in DTN normally focus on raising the average delivery ratio of the whole network but do not consider the fairness in the delivery performance among nodes. Thus their forwarding strategies is either to only spread messages not found in the next hop nodes to conserve bandwidth or to give higher priorities to messages that have higher chance of being delivered to the destination with the help of the encountered node. For example, in Epidemic and Spray and Wait, a node simply forwards all the messages which an encountered node does not have in its buffers yet. In Prophet and Maxprop, a node forwards a message to an encountered peer if the peer has a higher possibility to deliver the message to its destination node.

The forwarding and queueing policies above could be integrated into existing routing protocols as follows. For Epidemic and Spray and Wait, each node simply adopts the forwarding policy for the messages to be forwarded to the peer. That means the messages generated by nodes with high forwarding priority are forwarded first. For Prophet and Maxprop, denote d as the metric used to measure the routing capability of the peer for each message. Denote f as the forwarding priority a node gives to the source node of that message. The messages will be forwarded in the increasing order of $f \times d$.

5.4 Performance Evaluation

5.4.1 Simulation setup

The author simulates a sparse mobile network with a small number of nodes to achieve a scenario suitable for DTN. The general simulation setting is as follows. The simulation is implemented in ONE simulator [29]. There are 40 nodes with a transmission range

of 100 m. They travel over an area of 4500 m x 3400 m, at speeds of 10 km/h to 50 km/h, using the Shortest Path Map Based Movement Model available in ONE to simulate the movement of vehicles on the city streets. The simulation time is 43200s or 12 hours. The number of attackers is 12. The window size W is 20. This setting is applied to all the following experiments by default. For each experiment with distinct setting, the simulation is run for 10 times using random seeds and the average of the measured metrics is recorded.

The performance of FDER is assessed in detecting three types of misbehaviors: ER manipulation, message flooding and replica flooding. The following metrics are used for the evaluation purpose:

- Detection accuracy: percentage of malicious nodes that can be detected by normal nodes
- Detection delay: the average time taken to detect attackers among those normal nodes that can successfully detect the misbehavior.
- Wasted transmissions percentage: wasted transmission is defined as the normal node's transmission (relay or delivery) of a message generated by an attacker. Wasted transmission percentage is the ratio between the number of wasted transmissions and overall number of transmissions.

5.4.2 Detecting ER manipulation misbehavior

Attackers hold back unfavorable ERs to hide their dropping behaviors, which is the third strategy described in Section 5.2.3. The author considers this manipulation strategy because it is the most challenging to detect among three strategies. ER manipulation ratio (EMR) is defined as the ratio of the number of manipulated ERs

over all ERs for a malicious node. The attacker's EMR is varied from 0.01 to 0.2. The parameters α and β are set as 3 and 20 respectively.

Fig. 5.9 shows the detection rate and delay of normal nodes to detect attackers. The author finds that most of the malicious nodes can be accurately detected by all normal nodes when EMR is as high as 0.04. The detection delay reduces and the detection rate increases as EMR increases. This means the more frequently the attackers manipulate the ERs, the faster and more possibly they can be discovered.

Figure 5.9: ER manipulation detection time and accuracy.

5.4.3 Detecting message flooding

In this section, the author first evaluates the detection of message flooding when attackers reuse message count. The routing protocols used are Prophet, Maxprop and Spray and Wait. The time interval T is one hour. The rate limit M is 6 messages per T . The number of excess messages that each attacker sends per T is varied from 1 to 5. The author compares the detection performance of FDER with Li's scheme [66] in the same settings.

Fig. 5.10 plots the performances of FDER and Li's schemes in detecting the mes-

Figure 5.10: Performance of detecting message count reuse.

sage count reuse. When attackers send more superfluous messages per time slot, the detection delay reduces and the detection accuracy increases. As a result, the wasted transmission percentage decreases when attackers flood more messages. This means both FDER and Li's scheme can successfully control the malicious nodes' impact on the network. FDER performs slightly better than Li's scheme in all three metrics. This is because in FDER, all the sending activities of a node are compactly recorded in ERs and presented to other encountered nodes. Thus the inconsistency of message counts appearing in different encounters of the adversary could be detected very quickly. Meanwhile in Li's scheme, the message count records are randomly distributed

Figure 5.11: Performance of detecting burst-limit violation.

to different nodes and the inconsistency detection can only occur opportunistically.

Next the author evaluates the detection of message flooding when attackers exceed the number of valid burst slots Q . The settings used in the routing protocol, M and T , are as above. Furthermore, K is fixed as 5, Q is varied from 1 to 5. Each attacker is set to send more than M but less than or equal to $M + K$ messages per time slot for more than Q consecutive time slots. Attackers declare the correct message count. Though Li's scheme is not designed to enable burst-limit policy, the author applies Li's scheme to detect the burst-limit violation for the purpose of comparison.

Fig. 5.11 shows the performances of FDER and Li's scheme on detecting the violation of threshold Q . The general trend to observe in both schemes is that when the threshold Q increases, the detection delay and the wasted transmission percentage increases while the detection accuracy decreases. This is because when Q is larger, a node needs to collect the attackers' sending behavior over a longer time, which is more difficult and thus leads to the possibility that the node might not have enough information to conclude that it is indeed an attacker. It is seen that the detection accuracy of FDER is much higher than Li's scheme in this scenario. This proves the capability of ER to facilitate the monitoring of a node's sending behavior over multiple time slots.

Finally, the author evaluates the performance of the fairness policy by integrating it with the routing protocol Prophet. Each attacker is set to launch the maximum burst attack: it sends $M + K$ messages for consecutive Q slots, then sends M messages for one slot, and continues to alternate between these two patterns. The author fixes M as 6, Q as 4, T as 3600 s and varies K as 6, 12, 18, 24, 30. The average delivery ratio and delivery delay of normal nodes' messages are measured. The delivery ratio is the percentage of messages that is successfully delivered to the destinations over the total number of messages. The delivery delay is the time taken since the message is created until the time it is delivered to the destination.

Fig. 5.12 shows the average forwarding ratio of attackers compared to normal nodes over time. The forwarding ratios of attackers are degraded over time and become lower than those of normal nodes. Even if they might not be detected as malicious due to the maximum burst attack, their messages will be given less priorities compared to normal nodes. Fig. 5.13 shows the delivery performance of the network with and without the fairness policy (FP). When fairness policy is applied, the normal nodes' delivery ratios increase and their delivery delays reduce compared to

Figure 5.12: Forwarding priority of normal and malicious nodes over time.

(a) Forwarding priority of normal and malicious nodes over time.

(b)

Figure 5.13: Delivery performance of normal nodes with and without fairness policy (FP).

when FDER is not applied. Though when the number of superfluous messages per time slot increases (i.e. for more flexibility), the delivery performance degrades but insignificantly. Overall the fairness policy shows it works to help ensure the delivery performance of normal nodes even when the adversaries launch the maximum burst attack.

Figure 5.14: Performance of detecting replica count reuse.

5.4.4 Detecting replica flooding

The author evaluates the detection of replica flooding when attackers reuse replica count. The quota-based routing protocol called Spray and Wait is used. The initial number of copies L for a normal node is set to 5. The number of excess replicas per message of each attacker is varied from 1 to 5.

Fig. 5.14 plots the detection performances of FDER and Li's schemes with varying excess replica rate. When the number of superfluous replicas increases, the detection delay reduces and the detection accuracy increases. This is because the more replicas the attackers inject into the network, the higher is the possibility that they leave traces

Figure 5.15: FDER's detection of message count reuse attack when varying window size.

Figure 5.16: FDER's detection of replica count reuse attack when varying window size.

and get detected by the defence schemes. FDER's performance is higher than Li's scheme since the ER window in FDER helps nodes to quickly detect the inconsistency of replica counts assigned by attackers over different encounters.

5.4.5 Choosing window size

In this section, the author investigates the relationship between the window size W and the detection performance of FDER. For the attack of message count reuse, the superfluous messages per time slot is fixed at 3 and the window size is varied from 5

to 40. Fig. 5.15 shows the detection performances of FDER in this setting. As seen in Fig. 5.15(a), FDER can detect all malicious nodes accurately even if the window size is set to as small as 5. When the window size increases, the detection delay reduces as in Fig. 5.15(b), i.e. the detection performance is improved. This is because a larger window size contains more information of the malicious nodes' history and hence facilitate FDER to detect faster.

For the attack using replica count reuse, the superfluous replicas per time slot is fixed at 3 and the window size is also varied from 5 to 40. Fig. 5.16 presents the detection performances of FDER. From Figs. 5.16(a) and 5.16(b), it is observed that the optimal window size is 20, since it is the smallest window size that can maintain both a high accuracy and a relatively low detection delay compared to other values.

Figure 5.17: FDER's ER size.

The overhead of ER history storage and transmission is:

$$Overhead_{ER} = W \times size_{ER}$$

where W is the window size of ER history and $size_{ER}$ is the size of each ER. Fig. 5.17 shows an overview of the fields in the ER and their sizes in bytes. The size of each message record MR is 20 B. The author runs a simulation and obtains the average number of message records in SL (list of message records send by a node to another) and

RL (list of message records received by a node from another) as 20 for Prophet routing protocol. Thus, the average size of ER is $4 \times 5 + 20 \times 20 + 20 \times 20 + 40 \times 2 = 980\text{B}$. For the default window size W of 20, the total storage overhead of ER history is $20 \times 980\text{B} = 19.6\text{kB}$, which can also be comfortably afforded by mobile devices such as mobile phones and notebooks.

5.5 Conclusion

Flooding attack is a serious threat to wireless networks, including DTNs. In this chapter, the author proposes a distributed method FDER to detect flooding attack and allow flexible burst traffic in DTN. The simulation results show that the proposed approach can detect flooding attacks at high accuracy and low delay for different types of routing protocols. FDER outperforms the-state-of-art Li's scheme and incurs affordable overhead even in mobile devices like handphones. The fairness policy is shown to ensure the delivery performance of normal nodes even when attackers utilize the burst-limit policy to hijack more network resources for themselves.

Chapter 6

Balancing Trust and Privacy in VANET

In this chapter, the author proposes a secure and flexible framework for vehicles to manage both trust and privacy. The framework includes two components: ALRS (Adaptive Linkability and Recognition Scheme) and ATMS (Adaptive Trust Management Scheme) that work seamlessly to provide vehicles with the balance between trust and privacy. ALRS provides three levels of recognition among nodes despite them updating pseudonyms to protect privacy. The three levels include recognition by identity, by trust level and recognition revocation. Encryption and private set intersection technique are used to protect the linkability information and ensure the security in ALRS. ATMS is designed to adapt to the linkability information enabled by ALRS for vehicles to utilize and update entity trust as well as evaluating data trust. Simulation results demonstrate that the framework helps nodes to simultaneously make accurate decision towards the data and preserve their privacy in a flexible way.

6.1 Overview

6.1.1 Network model

It is assumed that VANET uses identity-based cryptography (IBC) system to provide authentication, encryption and privacy to the vehicles. In IBC, a trusted third party, called the Private Key Generator (PKG), is responsible for generating the private keys for nodes from their distinct identities. First, PKG publishes the master public key to the network but keeps the corresponding master private key. Any node can compute the public key corresponding to an identity based on the master public key and the identity value. However, only the node owning that identity can obtain the corresponding private key by requesting PKG, which uses the master private key to generate the private key for the identity.

Given a node A , denote its identity, the corresponding public key and private key as ID_A , P_A and S_A respectively. For the authentication service, node A signs the message with its private key S_A and attaches its identity ID_A with the message. Other nodes verify the signature with the public key P_A that can be computed from ID_A . For the encryption service, other nodes encrypt the message with A 's public key P_A so that only node A can decrypt to read the message using its private key S_A .

When a node uses the same identity all the time, its privacy is leaked to the eavesdroppers that correlate the identities to the locations. To protect nodes' privacy, the author adopts IBC pseudonym [84, 85] which is a pair of pseudo-identity and corresponding private key where pseudo-identity is a random value. Thus, different pseudonyms are uncorrelated to one another and uncorrelated to the real identity and nodes can avoid being traced when changing pseudonyms over time. PKG is in charge of issuing multiple pseudonyms to each node for use in the future. The node stores

these pseudonyms and uses one of them at a time for authentication and encryption purposes. To further enhance privacy, nodes can incorporate existing strategies such as [89–93] which propose proper locations and times for nodes to change pseudonyms.

Each node maintains a local reputation list of nodes it has encountered and the corresponding trust values. The trust value is quantified as a number in the range of $[0, 1]$. 0 means no trust and 1 means total trust. Denote TR_{AB} as the trust value of A towards B recorded in A 's local reputation system. When two nodes A and B first meet, they set the initial trust to each other with the value TR_{init} . If the trust value TR_{AB} drops below a threshold TH_{evil} , A considers B as malicious and isolates B by not sending or receiving any message from B .

6.1.2 Problem statement

When nodes use pseudonyms for privacy purpose, they cannot recognize one another, rendering it impossible to make use of entity trust. The goal is balancing between trust and privacy by designing a secure linkability scheme to enable legitimate nodes to recognize their encountered neighbors despite the pseudonym change. There are two requirements for the scheme. First, no linkability information is leaked to external attackers so that the linkability scheme does not affect the privacy level provided by the pseudonym scheme. In other words, the same privacy preservation is maintained against external nodes even when applying the linkability scheme. Second, linkability is flexible enough to allow internal nodes to adjust the privacy according to their wishes. In this way, even though the internal nodes might recognize their neighbors at times, they cannot trace other nodes all the time.

To achieve the first requirement, the author uses encryption to keep the linkability information confidential between the involved nodes. Even if external attackers can

eavesdrop data, they cannot read and extract any useful information. For the second requirement, the author defines three levels of linkability that a node A can grant to a peer B :

1. *identity linkability*: allowing B to recognize A 's identity
2. *trust linkability*: allowing B to recognize if A is trustworthy without the need to know A 's identity
3. *linkability revocation*: disallowing B to recognize A by both identity or trust level

Identity linkability, trust linkability and linkability revocation provide nodes with increased level of privacy in that order. By switching among these three recognition levels, nodes can adapt their privacy level to the contexts. Besides, the author defines two properties of linkability:

1. *selective linkability*: A can grant some linkability to selective peers while revoking that ability of other peers.
2. *asymmetrical linkability*: A may want to recognize B but does not want to be recognized by B

These properties provide more flexibility for nodes to adjust their privacy.

6.1.3 Basic idea of linkability scheme

Identity linkability can be achieved by a naive approach as follows. When two nodes encounter each other, they exchange identities confidentially with each other by encrypting its own identity with the peer's current pseudonym public key. However, this solution has several shortcomings. First, it incurs high communication cost as the pair

needs to explicitly and interactively resend their identities each time they meet and change pseudonyms. Second, it is hard to achieve trust linkability, linkability revocation, selective linkability and asymmetrical linkability, which are desired for a flexible linkability scheme.

To overcome these challenges, the author suggests that two nodes securely exchange their identities and agree on a unique shared secret value (called *link value*) for once only. Thereafter, they recognize the identity of each other by determining if the peer holds this link value. It is noted that the link value should be kept private between the two nodes only. Thus two nodes need to determine the common value between their lists of link values (which is corresponding to the peer's ID) without revealing other secret link values. This goal is realized with the cryptographic tool named Private Set Intersection (PSI) that allows two or more parties, each holding a set of inputs, to obtain intersection over their sets without leaking additional information.

Given two nodes A and B that have agreed on a link value L_{AB} . For asymmetrical linkability, the author defines two separate lists of link values for each node, let say node A : $idLinked_A$ involving nodes that A wants to be recognized by and $idLinking_A$ involving nodes that A wants to recognize. Node A recognizes node B 's identity by executing PSI between $idLinking_A$ and $idLinked_B$. Meanwhile, node B recognizes node A 's identity by executing PSI between $idLinked_A$ and $idLinking_B$.

For linkability revocation, a node cannot recognize a peer if the common value is not found between their lists of link values. Thus if node A wants to revoke node B 's ability to recognize its identity, node A simply removes L_{AB} from $idLinked_A$. Node A can re-enable node A 's recognition any time by adding L_{AB} back to $idLinked_A$. Selective linkability is thus achieved since A can independently choose to include or exclude specific link values in $idLinked_A$.

Figure 6.1: Recognition protocols executed by node B .

For trust linkability, the author needs to resolve the problem of how to let B learn if A is in B 's list of trusted peers, without knowing the exact identity of A . This goal is realized with Private Set Intersection Cardinality (PSI-CA) technique that allows two or more parties, each holding a set of inputs, to obtain the number of intersection values among their sets without leaking the actual intersection values and other items. Specifically, PSI-CA is executed between A 's list of link values and B 's list of link values associated with the trusted peers. If the number of intersection is 0, the link value L_{AB} is not in B 's trusted list, i.e. A is not trustworthy to B ; otherwise, A is among the peers B trusts. At the same time, B cannot learn the exact common value, hence cannot infer the peer's identity.

The author defines an additional list of link values for each node, e.g. A : $trustLinks_A$ involving nodes that A is willing to be recognized by trust level. Besides, node A classifies other nodes into three types: trusted, neutral and untrusted, and puts their corresponding link values into three respective lists: $highTR_A$, $midTR_A$ and $lowTR_A$. Node B recognizes node A 's trust level by applying PSI-CA on each of

the trust-level lists ($highTR_B$, $midTR_B$ and $lowTR_B$) with $trustLinks_A$. Node A revokes node B 's ability to recognize its trust level by removing L_{AB} from $trustLinks_A$.

In summary, Fig. 6.1 reviews how PSI and PSI-CA techniques work on the defined lists to enable node B to recognize node A by identity or trust level. Based on this, the author is going to design ALRS that satisfies the security and flexibility requirements stated in Section 6.1.2.

6.2 Preliminary

This section introduces the PSI and PSI-CA protocols used in ALRS.

6.2.1 PSI

Existing PSI protocols [97–101] use sophisticated cryptography to encrypt input items to prevent a malicious party from brute-force testing if an item appears in the other party's set. However, cryptography-based PSI might be expensive and unnecessary for high-entropy inputs (e.g. inputs generated randomly from a large domain) since enumerating sets in brute-force attack is impractical. Alternatively, recent works [102–104] propose PSIs using Bloom filter (BF) to present sets of high-entropy items before the sets intersection. BF-based PSIs rely on symmetric key operations and have linear complexity, thus they are more efficient, faster and more scalable than previous PSI protocols. At the same time, they ensure the security for high-entropy-items such as link values.

BF

BF [105] is a data structure used to efficiently represent set and test set membership. An empty BF is an array of s bits initially set to 0. BF uses k independent hash functions: $h_1, \dots, h_k$ with range from 1 to s . Given a set $X = (x_1, \dots, x_n)$ of n elements and a salt value r , denote BF_X as BF representation of X and $BF_X(u)$ as the bit value at position u in BF_X . The salt is used to add randomness to BF when necessary. BF_X is created by setting $BF(h_j(r||x_i))$ to 1 $\forall i = 1 \dots n, \forall j = 1 \dots k$. To test if an element y is a member of X , it is checked if $BF(h_j(r||y)) = 1 \forall j = 1 \dots k$. This function is referred as $testMem(y, BF_X, salt)$.

BFs introduce false positives, i.e., an element might seem present although it was never inserted. The false positive rate p can be approximated as:

$$p = \left(1 - \left[1 - \frac{1}{s}\right]^{kn}\right)^k \quad (6.1)$$

Given a number of items n and a desired false positive p , the optimal values of s and k for BF could be calculated as follows:

$$s = \text{ceil}\left(\frac{-n * \log(p)}{(\ln(2))^2}\right) \quad (6.2)$$

$$k = \text{round}\left(\ln(2) \times \frac{s}{n}\right) \quad (6.3)$$

BF-based PSI

Table 6.1 presents details of the mutual BF-based PSI [104]. Given two parties A with input set $\mathcal{A} = \{a_1, \dots, a_n\}$ and B with input set $\mathcal{B} = \{b_1, \dots, b_m\}$, the protocol allows both sides to learn $I = \mathcal{A} \cap \mathcal{B}$ securely. A and B respectively choose random salt values r_a and r_b and represent their sets as Bloom filters BF_A and BF_B (using the

Table 6.1: Bloom filter based PSI.

Input: $\mathcal{A}, r_a, BF_A$	Input: $\mathcal{B}, r_b, BF_B$
A	B
$\xrightarrow{r_a, BF_A}$	$\xleftarrow{r_b, BF_B}$
PSI ($\mathcal{A}, \mathcal{B}, \mathbf{r_b}$)	PSI ($\mathcal{B}, \mathcal{A}, \mathbf{r_a}$)
$I_{AB} = \emptyset$	$I_{BA} = \emptyset$
$\forall i = 1..n : \text{if}$	$\forall j = 1..m : \text{if}$
$\text{testMem}(a_i, BF_B, r_b)$	$\text{testMem}(b_j, BF_A, r_a)$
then $I_{AB} = I_{AB} \cup a_i$	then $I_{BA} = I_{BA} \cup b_j$
Output: I_{AB}	Output: I_{BA}

same k hash functions and number of bits s). A and B first exchange their BFs and salts, then execute $PSI()$ function to get intersection values. For example, A executes $PSI(\mathcal{A}, BF_B, r_b)$ which tests if each item in S_A appears in BF_B using $\text{testMem}()$. If an item gives a positive result, the item is deemed as a common value between their sets.

6.2.2 PSI-CA

Previous PSI-CA protocols [107–110] allow the operation between two or multiple parties with at least sub-quadratic complexity. The most efficient protocol [111] only requires linear communication and computation cost. This PSI-CA protocol is summarized in Tables 6.2 and 6.3. Given two parties A and B with respective input sets $\mathcal{A}$ and $\mathcal{B}$, PSI-CA protocol allows the client side B to learn $|\mathcal{A} \cap \mathcal{B}|$ only. The parameters and functions required by the protocol are described in Table 6.2. Compared to the original protocol in [111], the author introduces an additional parameter r which is the salt value chosen by node B to add randomness to the hash results. It is noted that

Table 6.2: PSI-CA: Parameters and functions

p, q	prime number
g	generator g of a subgroup of size q
R_a, R'_a	random numbers chosen by A : $R_a \leftarrow \mathbb{Z}_q, R'_a \leftarrow \mathbb{Z}_q$
R_b, R'_b	random numbers chosen by B : $R_b \leftarrow \mathbb{Z}_q, R'_a \leftarrow \mathbb{Z}_q$
r	random salt chosen by B : $r \leftarrow \mathbb{Z}$
x	input item
H	hash function $H : \{0, 1\}^* \rightarrow \mathbb{Z}_p^*$
H'	hash function $H' : \{0, 1\}^* \rightarrow \{0, 1\}^{\mathcal{K}}$ (where $\mathcal{K}$ is security parameter)
Π	permutation function on input set $\mathcal{X} = (x_1, \dots, x_p)$: $\Pi(\mathcal{X}) = (\hat{x}_1, \dots, \hat{x}_p)$
$f(x)$	$f(x) = H'(X^{R_b} \cdot H(x)^{R'_b})$ where $X = g^{R_a}$
$g(x)$	$g(x) = H(x)^{R'_a}$
$h(x)$	$h(x) = x^{R'_b}$
$k(x)$	$k(x) = H'(Y^{R_a} \cdot x^{1/R'_a})$ where $Y = g^{R_b}$

function $f(x)$ is equivalent to the composition of functions $g(x)$, $h(x)$ and $k(x)$:

$$f(x) = (k \circ h \circ g)(x) = k(h((g(x))))$$

This can be proven as follows. From the definition of parameters and functions:

$$(h \circ g)(x)^{1/R'_a} = ((H(x)^{R'_a})^{R'_b})^{1/R'_a} = H(x)^{R'_b} \quad (6.4)$$

$$Y^{R_a} = (g^{R_b})^{R_a} = (g^{R_a})^{R_b} = X^{R_b} \quad (6.5)$$

From Eqs. (6.4) and (6.5), it can be deduced that:

$$\begin{aligned}
(k \circ h \circ g)(x) &= H'(Y^{R_a} \cdot (h \circ g)(x)^{1/R'_a}) \\
&= H'(Y^{R_a} \cdot H(x)^{R'_b}) \\
&= H'(X^{R_b} \cdot H(x)^{R'_b}) = f(x)
\end{aligned}$$

Table 6.3: PSI-CA: Procedure (all computations are mod p).

A's input: $\mathcal{A} = \{a_1, \dots, a_n\}$	B's input: $\mathcal{B} = \{b_1, \dots, b_m\}$
<div style="display: flex; justify-content: space-between; align-items: center;"> <div style="width: 45%;"> $\forall i = 1..m : b'_i = h(b'_i)$ $(b''_{s_1}, \dots, b''_{s_m}) = \Pi(b'_1, \dots, b'_m)$ $Y = g^{R_a}$ $\forall j = 1..n : ta_j = f(a_j r)$ </div> <div style="width: 50%; text-align: center;"> <p>PSICA($\mathcal{B}, \mathcal{A}$)</p> $X = g^{R_b}$ $\forall i = 1..m : b'_i = g(b_i r)$ $\xleftarrow{r, X, \{b'_1, \dots, b'_m\}}$ $\xrightarrow{Y, \{b''_{s_1}, \dots, b''_{s_m}\}} \{ta_1, \dots, ta_n\}$ $\forall i = 1..m : tb_i = k(b''_{s_i})$ </div> </div>	
B's output: $ \mathcal{A} \cap \mathcal{B} = \{tb_1, \dots, tb_m\} \cap \{ta_1, \dots, ta_n\} $	

Table 6.3 illustrates details of the PSI-CA protocol. B first applies $g(x)$ on each of its set items concatenated with salt ($b_i || r$) and sends the results (b'_i) to A . Upon receiving, A applies $h(x)$ on them before shuffling them into b''_{s_i} and sending to B . Besides, A also applies $f(x)$ on each of its set items concatenated with salt ($a_i || r$) and sends the result (ta_i) to B . Finally, B applies $k(x)$ on b''_{s_i} into tb_i , i.e. $tb_i = (k \circ h \circ g)(b''_{s_i}) = f(b''_{s_i})$. B obtains the cardinality by counting the number of matches between the sets of ta_i and tb_i . Both A and B cannot learn the content of the other side's set as the items are masked with hashing and exponentiation functions. B cannot learn the matching value either since B cannot link b''_{s_i} back to the value of b_j , due to hashing and shuffling functions.

6.3 ALRS

In this section, the author presents ALRS to enable adaptive linkability as mentioned in Section 6.1.3. ALRS includes three protocols: identity linkability establishment,

identity recognition and trust recognition.

Table 6.4: Identity linkability establishment protocol.

A $\rightarrow$ GenBF (idLinked _A) 1. Compute Bloom filter I_A for $idLinked_A$ using salt r_A 2. Attach (r_A, I_A) to the beacon.
B $\rightarrow$ IdRecognize (r _A , I _A) 1. Compute $I = PSI(idLinking_B, idLinked_A, r_a)$ 2. [if $I \neq \emptyset$], B determines peer's identity as A linked to $l \in I$ 3. [if $I = \emptyset$], B may request A to establish linkability.

6.3.1 Identity linkability establishment protocol

Two nodes A and B can establish the identity linkability in two ways: offline and online. If A and B already know each other in advance (for example, friends, familiarities in real life or online social network...), they can establish the linkability offline before they meet each other on the road. If they are unknown to each other, they can establish it online upon an encounter on the road for the first time.

Table 6.4 illustrates the protocol on how A establishes identity linkability with B . Denote ID_A , P_A , S_A as the real identity and corresponding public and private keys of node A . Denote a , p_a , s_a as the pseudo-identity and the corresponding public and private keys currently used by node A . The same notation meaning applies for ID_B , P_B , S_B , b , p_b and s_b of node B . First A chooses a link value L_{AB} and encrypts it with the peer pseudo-identity b . A attaches the encrypted value and its signature over the request before sending it to B . B replies to A if B agrees to exchange the real ID and establish the link value or not. If B agrees, A and B send their own identities that are encrypted with the other party's current pseudo-identity. If the authentication succeeds, A and B can record the link between L_{AB} and the peer's identity.

The linkability establishment may fail if B agrees to send its identity to A but B fails

to receive A 's identity. This is due to either A and B are out of the transmission range when A sends the reply or A wants to cheat B to know its identity. As the information of B 's identity is encrypted with A 's public key pseudonym, external attackers cannot obtain B 's identity to carry out tracking attack. Though A knows B 's identity, A cannot always recognize B in later encounters. To enhance its privacy, B may change its pseudonym to prevent A from recognizing B based on pseudonym or reject A if it requests B for linkability establishment again.

6.3.2 Identity recognition protocol

Table 6.5 shows the identity recognition protocol in which two nodes A and B attempt to identify each other. Given $idLinking_A$ and $idLinked_A$ as the lists of link values that A wants to recognize and to be recognized respectively. The same notation applies to $idLinking_B$ and $idLinked_B$. A first chooses the salt r_A as the current pseudo-identity value and then computes BF for $idLinked_A$ using r_A . The salts change ensures that node's BFs are varied and unlinkable when nodes change pseudonyms. A attaches this BF to the beacon and broadcasts it. Upon receiving A 's beacon, the neighbor node B executes PSI to get the intersection value between $idLinked_A$ and $idLinking_B$, thereby inferring A 's identity.

Revoking identity linkability

Node A can revoke and resume node B to recognize it by removing or adding L_{AB} to $idLinked_A$ respectively. Thus the revocation operation is non-interactive and can take place any time a node wants. The revocation affects the linkability establishment protocol as follows. Suppose A and B has already established the identity linkability. There are two cases of revocation: (1) A and B both disable linkability, (2) A enables

Table 6.5: Identity recognition protocol.

NOTATION $ENC(m, ID)$: encryption of m with identity ID $DEC(c, S)$: decryption of c with private key S $SIG(m, S)$: signature over m with private key S , $VER(m, ID)$: verification of signature with identity ID	
A : real identity ID_A (public key P_A , secret key S_A), current pseudonym: a (public key p_A , secret key s_A)	B : real identity ID_B (public key P_B , secret key S_B), current pseudonym: b (public key p_b , secret key s_b)
A $\rightarrow$ ReqIdLinkability(p_B) 1. Select random value L_{AB} 2. Create request: $e_1 = ENC(L_{AB}, b)$, $r_1 = (a b e_1)$, $s_1 = SIG(r_1, s_A)$	
$(r_1 s_1)$	B $\rightarrow$ RepIdLinkability(p_A) 1. Verify signature $VER(s_1, a)$ 2. Decrypt $L_{AB} = DEC(e_1, s_b)$ 3. [agree] Create reply: $e_2 = ENC((L_{AB} ID_B), a)$ $r_2 = (b a e_2)$, $s_2 = SIG(r_2, s_b)$, $s_3 = SIG(r_2, S_B)$
$(r_2 s_2 s_3)$	
3. Verify signature $VER(s_2, b)$ 4. Decrypt $DEC(e_2, s_a)$ to get L_{AB}, ID_B 5. Verify signature $VER(s_3, ID_B)$ 6. Create reply: $e_3 = ENC((L_{AB} ID_A), b)$, $r_3 = (b a e_3)$, $s_4 = SIG(r_3, s_a)$, $s_5 = SIG(r_3, S_A)$	4. Verify signature $VER(s_4, a)$ 5. Decrypt $DEC(e_3, s_b)$ to get L_{AB}, ID_A 6. Verify signature $VER(s_5, ID_A)$
$(r_3 s_4 s_5)$	
Record linkability: (L_{AB}, ID_B)	Record linkability: (L_{AB}, ID_A)

and B disables the linkability. It is assumed that the former case occurs when both A and B want to keep their respective privacies with other internal nodes. Thus even when they fail to recognize each other, they do not bother to establish the new linkability. In the later case, A cannot recognize B but B can recognize A and knows that it currently wants to hide its identity from this peer. A may request to establish the linkability but B can choose to reject it to protect its privacy from A .

Reducing false positive rate of BF

BF-based PSI enables the author to construct a lightweight, fast, scalable and non-interactive identity recognition protocol. However, BF may incur false positive in intersection results of PSI. Based on Eq. 6.1, a direct way to reduce the false positive rate is to increase the number of bits s and number of hashes k of BF. To achieve a negligibly small false positive, the BF size could be very huge, incurring high communication overhead. [106] proposes a more efficient way to reduce false positive and maintain the BF size simultaneously, which is using additional BFs of different hashes to cross-check on the positive result of the main BF; if any cross-check produces a negative, the positive of the main BF is determined as a false positive. Applying this idea on the recognition protocol, the author proposes that each node creates t BFs for the same list of link values using different salt values. The node can attach all t BFs on the same beacon or attach a different BF on each beacon. The neighbors apply PSI on these BFs to cross-check if a positive result is false or not. Section 6.5.3 will further elaborate the efficiency of this scheme in reducing false positive.

6.3.3 Trust recognition protocol

For a node A , denote $highTR_A$, $midTR_A$ and $lowTR_A$ as the respective lists of peers that A considers as trusted, neutral and untrusted. The classification of trust-level is based on the local reputations maintained by A . If the trust TR_{AB} of A towards B is above the threshold TH_{friend} , A puts B into $highTR_A$. If TR_{AB} is below the threshold TH_{evil} , A puts B into $lowTR_A$. Otherwise (i.e. the trust falls between TH_{friend} and TH_{evil}), B is put into $midTR_A$.

Suppose a node B cannot recognize the identity of the peer A after executing $IdRecognize()$ protocol. If B wants to know A 's trust level (for example, to

Table 6.6: Trust recognition protocol.

B $\rightarrow$ **TrustRecognize**(**highTR_B**, **trustLinks_A**)

1. Compute $c_1 = PSICA(highTR_B, trustLinks_A)$
 $c_2 = PSICA(midTR_B, trustLinks_A)$
 $c_3 = PSICA(lowTR_B, trustLinks_A)$
2. [if $c_1 > 0$], B determines peer is trusted
3. [if $c_2 > 0$], B determines peer is neutral
4. [if $c_3 > 0$], B determines peer is untrusted

evaluate some data involving A 's opinion), B applies the PSI-CA protocol on each of the trust-level lists ($highTR_B$, $midTR_B$ and $lowTR_B$) with $trustLinks_A$. Table 6.6 shows the trust recognition protocol that requires three times of executing PSI-CA: $PSICA(highTR_B, trustLinks_A)$, $PSICA(midTR_B, trustLinks_A)$ and $PSICA(lowTR_B, trustLinks_A)$. As seen in Table 6.3, each PSI-CA takes place in two rounds of interaction between two nodes. If A and B implement PSI-CA for three times consecutively one after another, six rounds of interaction are incurred. To reduce the interaction overhead, the author proposes that each node calculates the data for the respective three PSI-CA calls and integrates them into one packet to send at a time. In this way, the number of interaction rounds remains as 2. It is noted that node B should choose a different salt to execute PSI-CA when it updates pseudonym. This ensures the hash results of its list of link values change accordingly and become unlinkable over time.

6.3.4 Security analysis

ALRS is secure as long as the three protocols presented above are secure. In the identity linkability establishment protocol, two vehicles exchange their real identities and a shared link value which is encrypted with the peer's current pseudonym. Thus, these private information are kept confidential from external attackers and other unrelated

internal nodes. The security of the identity recognition and trust recognition protocols respectively relies on the security of the PSI and PSI-CA protocols presented in Section 6.2. [104] proves that the BF-based PSI protocol only lets the involved parties learn the common items but not other items in the other party's set. Applying BF-based PSI on high-entropy items of link-values is robust against the brute-force attack. Thus the identity recognition protocol ensures the nodes' identity linkability and linkability revocation are secure. [111] proves that the proposed PSI-CA protocol only lets the client party learn the number of common items without leaking other information such as the value of common items and other items in the set to both parties. Exceptionally, the client can infer the values of common items if the size of the client set is equal to the number of common items. However, this exception rarely happens for high-entropy items of link values. Besides, the brute-force attack to guess the common link values incurs significant overhead to enumerate all possible sets of link values and execute the interactive PSI-CA protocol for multiple times. Thus the trust recognition using this PSI-CA protocol can robustly provide nodes' trust linkability without revealing the real identities. It is noted that both PSI and PSI-CA use varied salt to prevent unlinkability when nodes execute these protocols multiple times on the same list of link values. By changing salts upon pseudonym update, vehicles can avoid being traced when they run the recognition protocols.

ALRS can function despite pseudonym change and does not regulate when and where vehicles are allowed to change pseudonyms. Thus, ALRS allows vehicles to enjoy the same level of privacy against attackers as the original pseudonym schemes and pseudonym-update strategies.

6.4 ATMS

Making use of ALRS, the author proposes ATMS to permit vehicles to evaluate the trustworthiness of data despite that vehicles apply pseudonyms to protect their privacy. The local danger warning (LDW) [15] application is used as the concrete scenario to design ATMS. In LDW, vehicles generate and disseminate event messages to inform other vehicles in the vicinity about certain dangers on the road. ATMS allows vehicles to evaluate if the received event message is trustworthy or not, using the entity trust provided by ALRS.

ATMS includes three modules: objective evaluation of event, subjective evaluation of event and update of peers' reputation. A vehicle evaluates the credibility of event message in either objective or subjective manner, depending on its context. If the vehicle is near enough to the danger to observe directly, it generates and broadcasts an event message. For the vehicle that is a direct neighbor of the nodes reporting event messages, it can objectively observe their movement responses to the reported danger. The message is believable if the source node's response is consistent with its opinions towards the danger. For the vehicle that stays too far to directly observe the events and the responses, it has time to collect the opinions of different source nodes towards the danger. Thereby it can subjectively evaluate the event by aggregating those opinions in consideration of their entity trusts which are set adaptively to the granted linkability in ALRS. Finally, the vehicle utilizes the evaluation results to update its trusts towards the messages' source nodes.

First, Section 6.4.1 describes LDW and the attack model against LDW. Next, the modules of ATMS are presented respectively in Sections 6.4.2, 6.4.3 and 6.4.4. Finally, Section 6.4.5 discusses the trade-off between trust and privacy when using ATMS and ALRS together.

6.4.1 Application scenario: LDW

When drivers are travelling, they may encounter hazards on the road such as possible cars collision, debris or ice, etc. Drivers can detect these hazards manually by observations with the eyes or automatically with on-board sensors on their vehicles (e.g. radar, lidar). However, both eyesight and traditional sensors are limited in range. To broaden the scope of danger perception, vehicles can utilize communication to propagate warnings about the danger in the vicinity. This goal is realized in LDW application [15].

In LDW, a vehicle that detects nearby hazards on the road using its on-board sensors will generate a warning and broadcast it. Vehicles that receive the warning will relay it to other vehicles. If a vehicle is approaching the potential dangerous area, it needs to evaluate the situation and decide on taking actions or not.

Fig. 6.2 illustrates three areas associated with a local danger. The detection area is the innermost region where vehicles can detect the presence or absence of hazards using their sensors. The decision area is the region where vehicles should decide to take actions or not to avoid the danger. When vehicles enter the dissemination area, the vehicles start collecting and distributing warnings. These areas are illustrated as circles but their actual shapes and sizes depend on the scenario factors such as hazard type and road structure.

There are two types of messages in LDW application: event and beacon. A vehicle either generates or relays an event message which conveys the warning of potential danger on the road. From now on, the terms "danger" and "event" might be used interchangeably. Besides, each vehicle periodically broadcasts a beacon message which specifies its status, including information such as its location and velocity.

Figure 6.2: Areas associated with local danger.

Event message

When a vehicle A is in the detection area of a hazard, it generates an event message m and broadcasts m to inform neighbor nodes about the event:

$$m = (E, X, Y, p, l, t, op)$$

where E is the event type (such as possible collision, accident on road, dangerous road condition...), X and Y are the coordinates of the event's location, p , l and t are the pseudonym, location and time of vehicle A when it generates m . op denotes the opinion of node A towards the event. op can take either of two values: 1 if A affirms the existence of the event and -1 if A negates it. In the case A honestly reports an actual danger, op is set to 1.

When a vehicle B receives m (from source node or other relays), it re-broadcasts m if this is the first time B receives m and B is within the dissemination area of the event. If B later discovers that the event is non-existent, B generates a negation message (where op is set to -1) and broadcasts it.

Beacon message

The beacon message b of a vehicle A is formatted as:

$$b = (p, x, y, s, \theta, t)$$

where p , (x, y) , s , θ and t are the pseudonym, coordinate location, speed, driving direction angle and time of A when it generates b .

Attack model

LDW is threatened by the internal adversaries that generate fraudulent event messages. If the event exists, the adversaries generate and send the negation event messages. If the event does not exist, they generate and send the affirmation event messages.

6.4.2 Objective evaluation of event

Suppose node A generates an event message m and node B directly receives m from A . If A has established the linkability with B and enables this identity linkability, B can recognize A even if A changes pseudonym after sending m . Thus B can collect several subsequent beacons of A and judge m based on the objective evaluation of A 's beacon validity and subsequent movement. In case B cannot acquire A 's subsequent beacons, it can depend on the subjective evaluation as in Section 6.4.3.

The objective evaluation of event message m involves node B performing the following checks:

- (a) *location of source node*: checking if the location of the source is within the detection range of the event location

- (b) *freshness of message*: checking if the message timestamp is within the expiration period
- (c) *beacon validity*: verifying A 's beacons collection as in Section 6.4.2.
- (d) *subsequent movement*: checking if A 's subsequent movement (speed, location) is a valid response to the event reported in m . If the danger is present, A should slow down and avoid the location of the danger. Otherwise, the danger warnings may be wrong.

Verifying beacon message

Suppose node B has obtained k consequential beacons of node A : $b_1, \dots, b_k$ where $b_i = (p_i, x_i, y_i, s_i, \theta_i, t_i)$ for $i = 1 \dots k$. Node B verifies these beacons with two tests: threshold and tracking test. The former test involves checking the speed, location and timestamp claimed in individual beacon against valid thresholds. The latter test aims at verifying the consistency of each beacon with one another.

For each beacon b_i , node B performs the threshold test as follows:

- (a) *speed test*: checking the speed s_i against the allowed maximum speed S_{max} which depends on the road type. For example, S_{max} could be set to 50 km/h for urban, 100 km/h for highway.
- (b) *range test*: checking if the location (x_i, y_i) is within the transmission range R_{max} of B . For the wireless standard used in VANET, R_{max} can be up to 300 meters.
- (c) *timestamp test*: checking if the timestamp t_i expires or indicates a point of future time.

If all the beacons pass the threshold test, node B proceeds with the tracking test as

follows. For each pair of subsequent beacons b_i and b_j , node B checks if the following condition is violated:

- (a) *beacon frequency test*: checking if the frequency of beacon generation exceeds the threshold F_{max} , i.e. if $\Delta t = t_j - t_i > F_{max}$.
- (b) *speed change test*: checking if the acceleration $|(s_j - s_i)/\Delta t|$ exceeds the threshold A_{max}
- (c) *direction change test*: checking if the driving direction change rate $|(\theta_j - \theta_i)/\Delta t|$ exceeds the threshold D_{max}
- (d) *location consistency test*: testing the similarity between the location specified in beacon b_j and the location at time t_j expected from beacon b_i . If A 's location at time t_i is (x_i, y_i) , its expected location at time t_j is:

$$x' = \Delta t \times s_i \times \cos\theta_i + x_i$$

$$y' = \Delta t \times s_i \times \sin\theta_i + y_i$$

b_i and b_j are correlated if (x', y', s_i) and (x_j, y_j, s_j) have high cosine similarity:

$$C(b_i, b_j) = \frac{x' \times x_j + y' \times y_j + s_i \times s_j}{\sqrt{x'^2 + y'^2 + s_i^2} \times \sqrt{x_j^2 + y_j^2 + s_j^2}}.$$

If $C(b_i, b_j) > C_{min}$, b_i and b_j are deemed as on the same track and judged as valid.

6.4.3 Subjective evaluation of event

Consider node B receives event message m (either directly from message's source or through relays) but lacks information to objectively evaluate m due to the far distance from the event or the disabled identity recognition. Instead, node B can evaluate the

message subjectively by aggregating the opinions of different nodes towards the event, taking into account of those nodes' trust levels.

Suppose node B receives reports towards the same event from k source nodes: $A_1, A_2, \dots, A_k$. For $i = 1 \dots k$, op_{A_i} is denoted as the opinion of each node A_i towards the event: $op_{A_i} = 1$ if A_i affirms the event exists and $op_{A_i} = -1$ if A_i negates it. t_{BA_i} is denoted as the entity trust of B towards A_i . In the context of privacy where nodes vary pseudonyms over time, it is not trivial for node B to obtain t_{BA_i} . The author is going to describe in detail how to use linkability property in ALRS to set the proper value for t_{BA_i} in Section 6.4.3.

The author proposes the weighted majority voting for B to evaluate the event where the weight is equal to the trust value towards the event's source node. The aggregated trust towards the event is: $T = \sum_{i=1}^k t_{BA_i} \times op_{A_i}$. The danger event is evaluated as existent if $T > 0$ and non-existent if $T < 0$. When the vehicle enters the decision area of a reported event, it depends on the evaluation result to take action. If the evaluation concludes that the event exists, it slows down. Otherwise, it remains at the original speed.

Setting entity trust

Consider the scenario that node B receives the event message m generated by node A and needs to infer the entity trust t_{BA} towards A to accommodate subjective evaluation of m . Since A uses pseudonyms, B cannot easily recognize A 's identity and directly look up the entity trust in B 's local reputation system. However, B still can set t_{BA} with a proper value adaptive to the linkability enabled by A and B 's neighbors. The author addresses how B achieves this in two cases: (1) B and A are neighbors and (2) B and A are not neighbors.

In the former case, B can obtain A 's beacon since they are direct neighbors. B first executes the identity recognition protocol on A 's beacon to identify A . If A enables identity linkability with B , B can recognize A and get TR_{BA} directly from its local reputation system. Otherwise, B proceeds with requesting A for trust recognition. If A enables trust linkability with B , B can recognize A 's trust level. Given that B defines $TR_{trusted}$, $TR_{neutral}$ and $TR_{untrusted}$ as respective default trust values for trusted, neutral and untrusted peers, B can set t_{BA} to the default value corresponding to A 's trust level. If B cannot recognize both identity or trust level of A , B considers A as a stranger and set t_{BA} to the default value TR_{init} .

In the latter case where B is not a direct neighbor of A , B can neither recognize A 's identity nor request A for trust recognition. The author fixes this issue by requiring source node A to attach the salt value s_A and the Bloom filter BF_A for $idLinked_A$ to the original event message m . The new format of the event message generated by A is $M = (m || BF_A || s_A)$. Since the identity recognition protocol is non-interactive, B can apply it on the provided BF to recognize A 's identity despite the fact that node B is several hops away from A . If A enables identity linkability with B , B sets t_{BA} to TR_{BA} .

If B cannot recognize A 's identity, B should not initiate the trust recognition request with A . Since this protocol requires two rounds of interactive communication between nodes, executing it is costly for A and B which are several hops away from each other. Instead, the author suggests transitive trust approach in which B aggregates the trust value of B 's direct neighbors towards A . Suppose node D relays the event message M generated by node A . If node D can recognize A by identity or trust level, D is required to attach the obtained trust value t_{DA} to the message M before relaying. Otherwise, node D specifies the attached trust as *NULL* value. The new format of the event message relayed by D is $R = (M || t_{DA})$. Suppose node D is one of B 's direct

neighbors that relays event message M to B , B can obtain the transitive trust value towards A if t_{DA} is not $NULL$:

$$TT_{BA} = t_{BD} \times t_{DA}$$

It is noted that t_{BD} is obtained as specified in the case two nodes are neighbors. If there is no neighbor providing transitive trust, B will set t_{BA} to the default value TR_{init} . If B has k neighbors $D_1 \dots D_k$ providing their trust values $t_{D_1A} \dots t_{D_kA}$, B takes the average value of these transitive trust values as the final value for t_{BA} .

Figure 6.3: Aggregating transitive trust.

Fig. 6.3 exemplifies how node B aggregates available transitive trust values from three neighbors D_1 , D_2 and D_3 . As D_3 has no trust information towards A , B only considers the attached trust of D_1 and D_2 . Fig. 6.4 summarizes the setting of entity trust adapted to the conditions of neighbor relationship and enabled linkability.

Figure 6.4: Decision tree of linkability-based entity trust.

6.4.4 Update of peers' reputation

When node B reaches the observation range of the reported event, B can directly verify if the event exists. Accordingly, B can update the reputations for the source nodes of the event messages if B can recognize their identities. If the identifiable source node A reports honestly, B increases the reputation for A : $TR_{BA} = \max\{TR_{BA} + \alpha, 1\}$. Otherwise, B decreases the trust in A : $TR_{BA} = \min\{TR_{BA} - \beta, 0\}$.

The author also applies the ageing process for trust reputation update in which B reduces the reputation of A if B has no interaction with A for a long time. Denote the ageing factor as γ and the ageing period as τ . If B has no interaction with A for a time period of t , the reputation of A is updated as: $TR_{BA} = \min\{TR_{BA} \times \gamma^{t/\tau}, 0\}$. The ageing process encourages nodes to enable id linkability when possible so that their reputations can be increased. It also mitigates malicious nodes from the following misbehavior: first well behave, enable id linkability with other nodes and earn high trust from them, then misreport events and enable trust linkability only to cheat other nodes to believe in their reports.

6.4.5 Balancing trust and privacy

As sensitive information such as nodes' identities are encrypted before sending in ALRS, the proposed framework ensures that vehicles achieve privacy against external eavesdroppers when using pseudonyms and ALRS together.

ATMS depends on the privacy level that vehicles choose in ALRS as shown in Fig. 6.5. If A grants identity linkability to B , B can use A 's entity trust for any purpose (objective, subjective evaluation of event and trust update). If A grants trust linkability to B , B can use A 's trust level to make more informative subjective evaluation of data but cannot update A 's trust. If A revokes B , A appears as a stranger to B with no trust information. It is noted that in the case of trust linkability and revocation, B still might be able to evaluate the data objectively if A does not change the pseudonyms after sending the event message.

There exists a trade-off between the vehicles' privacy against other internal nodes and trust management: the higher the level of privacy, the less is the information available to the trust management and vice versa. However, this trade-off is less serious compared to the trade-off between the vehicles' privacy against external eavesdroppers and trust management. Besides, the proposal compensates the trade-off by allowing vehicles to flexibly adjust privacy by enabling or disabling the linkability. Vehicles do not need to enable linkability all the time, but they should do it when necessary (for example, in the presence of an event report).

Figure 6.5: Relationship of trust and privacy

6.5 Performance Evaluation

6.5.1 Setup

The author simulates VANET scenario in ONE simulator with the settings as follows. There are 300 nodes with transmission range of 250 m. They travel over an area of 5 km x 5 km, at speeds of 10 km/h to 50km/h, using the Shortest Path Map Based Movement model available in ONE to simulate the movement of vehicles on the streets. The simulation time is 43200 s or 12 hours. The author varies two parameters: (i) the number of attackers, (ii) the probability that each node enables id and trust linkability with other nodes. For each experiment with distinct setting, the author runs the simulation for 10 times using random seeds and the average of the measured metrics is recorded.

The author assesses the performance of ATMS with two metrics: decision accuracy and detection accuracy. Decision accuracy is the percentage of correct decisions that a node makes when it enters the decision area of a reported event. Detection accuracy

is the percentage of malicious nodes that can be detected by normal nodes.

6.5.2 Results

Varying number of attackers

For the first experiment, the author varies the number of attackers from 25 to 250 while fixing the probability of enabling id linkability to 100%. Fig. 6.6 shows the correct decision rate of the framework and a related work SBAP [96] when the number of attackers are varied. It is observed that the more the attackers, the lower is the correct decision rate. This is because more attackers give more weights to the wrong opinion, leading to the node's misjudgement when it is surrounded by more attackers than normal nodes. However, ATMS still achieves high correct decision rate of at least 60% when the number of attackers is 250. The framework always outperforms SBAP due to the combination of the objective and subjective evaluation.

Fig. 6.7 shows the detection rate of malicious nodes in the framework when the simulation ends. The detection rate is 50-70% when the number of attackers is varied.

Figure 6.6: Correct decision rate.

Figure 6.7: Detection rate of malicious nodes.

Varying probability of linkability

For the second experiment, the number of attackers is fixed at 150. The author varies the probability that a node enables id linkability from 0.4 to 1.0 and fix trust linkability probability at 0.0. The result achieved by the proposal in this setting is illustrated in Fig. 6.8. When the probability of id linkability varies from 0.4 to 1, the correct decision rate increases from 0.55 to 0.86. The higher the probability of id linkability, the higher is the decision rate since id linkability allows nodes to evaluate events more accurately.

Figure 6.8: Correct decision rate under varying probability of enabling id linkability.

For the third experiment, the number of attackers is still 150. The author fixes the probability that a node enables id linkability at 0.4 while varying the trust linkability probability from 0 to 0.6. Fig. 6.9 shows that the correct decision rate increases from 0.55 to 0.7 when the probability of trust linkability increases from 0.0 to 0.6. It is observed that trust linkability improves the performance of decision.

Figure 6.9: Correct decision rate under varying probability of enabling trust linkability.

6.5.3 Overhead

The overhead of the framework depends on the overhead of PSI and PSI-CA protocols presented in Section 6.2. For the PSI protocol, denote n as the average number of items (i.e. link values in the list for intersection), s and k as the number of bits and hash functions used to construct BF. The communication overhead is s bits and the computation overhead is $k \times n$ hash operations. Since n is in the scale of number of nodes N in the network (i.e. $O(N)$), set n as 300. Based on Eqs. 6.2 and 6.3, the author can obtain the false positive p to as low as 10^{-5} by setting s as 7189 bits (or 899 B) and k as 17. As a result, the communication overhead is 899 bytes and the computation overhead is 5100 hash operations.

The false positive rate could be reduced without increasing s by using t varied-salt

BFs to verify the positive result as described in Section 6.3.2. In this way, the false positive is reduced to p^t . The communication overhead is increased by t times if a node attaches all t BF's to each beacon, and the computation overhead is increased by a few hash operations to verify the positive result only. To achieve the same false positive rate of p^t by simply changing the parameters of BF, both s and k need to be increased by t times, consequently making both computation and communication overhead increase by t times. Thus, the cross-check approach is more efficient in reducing BF's false positive. The communication overhead could be further reduced by letting a node broadcast a different BF per beacon at the cost of delaying the positive verification for a few beacon periods. For example, if t is set to 2, the false positive is 10^{-10} , the communication overhead is still 899 B and the computation overhead is 10200 hash operations.

The overhead of the PSI-CA protocol is linearly proportional to the size of set which is $O(n)$. Specifically, the computation overhead for each node is approximately $2n$ exponentiations and n multiplications, which is about 900 operations in total for $n = 300$. The communication overhead amounts to about $2n$ $|p|$ -bit and n $\mathcal{K}$ -bit values. In practice, $|p|$ is selected as 1024 or 2048, and $\mathcal{K}$ depends on the hash function (for example, 128 bits for MD5, 160 bits for SHA-1 and 224, 256, 384 or 512 bits for SHA-2). For example, if $|p|$ is set as 1024 and $\mathcal{K}$ is set as 224, the communication overhead is about 681600 bits (or 85.2 kB).

6.6 Conclusion

To deploy VANET in practice, it is necessary to consider both issues of trust management and privacy protection which unfortunately require a trade-off between them. To resolve this trade-off, the author proposes a secure linkability protocol that enables

vehicles to recognize the identities or trust levels of other vehicles to facilitate the entity trust management. Different levels of recognition permit vehicles to maintain their privacy flexibly. The author also designs a context-aware trust management scheme that works seamlessly with the linkability protocol to allow vehicles to make more informative evaluation of the received data. Simulation results demonstrate that the proposed framework can achieve accurate trust evaluation in the privacy context, thus balancing the trade-off between them.

Chapter 7

Conclusions and Future Work

Early works on DTN and VANET only focus on improving the functionalities of these networks without considering security and privacy despite that these issues play a role in determining the usability of the networks. In this thesis, the author has addressed different kinds of security and privacy problems in DTN and VANET, which in turn enhances the operation robustness of these networks. In DTN, due to the lack of connectivity resources and end-to-end path information, it is challenging to detect the neighbor discovery and routing misbehavior of mobile nodes. To resolve the issue, the author has proposed the distributed use of either infrastructure nodes or ERs to monitor the behaviors of nodes over the network. Though these resources incur extra cost, the overhead is kept affordable while providing DTN with more security. Due to the involvement of real vehicles, VANET requires not only trust management but also privacy protection. The author has proposed a secure framework for vehicles to manage privacy and trust adaptively according to the context of operation. As a result, the balance between security and privacy is provided and the trade-off between them is mitigated. In the following, the author concludes the thesis by examining the benefits and trade-offs of the proposals and suggests a few directions for future work.

7.1 The Enhancement of Security and Privacy for DTN and VANET

7.1.1 Security against wormhole attack

In chapter 3, the author explores the statistical approach to detect wormhole attack in DTN. The approach based on time and location often relies on extra hardware equipment on each node, incurring high cost. The approach based on graphs exploit certain topologies formed by nodes, which might not be available in a sparse network like DTN. Focusing on the statistical approach, the author needs to resolve the challenge of discovering a statistical parameter as an indicator of the attack which is independent on the network density.

The author has found out that the neighbor count change in a small area is the potential solution to address wormhole. The analysis shows that this parameter works in DTN while other parameters suggested in previous works fail in such sparse network scenario. Exploiting this observation, the author has proposed SNDW which involves infrastructure nodes to monitor the statistical change of neighbor counts in the assigned areas, thereby detecting the wormhole attack.

The main insight of SNDW is that the neighbor count parameter works independently of network connectivity or nodes' speeds. Therefore, SNDW achieves the detection performance with higher accuracy and lower false positive than the prohibited-topology method in DTN. The disadvantage of SNDW is the requirement of a few infrastructure nodes to monitor the network. However, the number of infrastructure nodes could be reduced by piggybacking on existing infrastructure and complementing with optimised infrastructure layout and optimised monitoring schedule.

7.1.2 Security against blackhole and greyhole attacks

In chapter 4, the author studies the blackhole and greyhole attacks in DTN. Detecting the message dropping misbehavior is challenging in DTN since the data are routed opportunistically and do not follow predefined paths. Thus no single node can monitor all the sending activities of other nodes and there is no central server to control the network traffic either. The author has proposed ER – a secured record of the message exchange during an encounter between nodes, as the tool for nodes to learn about other nodes' sending and receiving histories in a distributed manner.

The author has designed SDBG that relies on the statistical metrics inferred from the nodes' ER histories to detect both collusion and individual attacks. For individual attack, low relaying ratio and high self-forwarding ratio are used as indicators of dropping misbehavior. For collusion attack, the abnormally high frequency of encounters and high number of sent messages between nodes are used as to filter colluders. Nodes that exhibit the above symptoms will have their reputations reduced. When the reputations fall below a threshold, they are isolated as malicious nodes.

Incorporating the statistical approach and the reputation system, SDBG can detect collusion and individual attacks at higher accuracy and much lower false positive than the related work of Li. Setting the proper values of reward and punishment for the reputation system makes SDBG robust against the attack that switches between good and adversary behaviors. The security of SDBG is achieved at the cost of maintaining and storing ER history. However, the author has shown that this overhead is affordable and adjustable to fit the node's performance requirements and constraints.

7.1.3 Security against flooding attack

In chapter 5, the author explores the use of ER to address another security problem in DTN, named flooding attack. Different from previous works, the author provides nodes with the capability to transmit messages more flexibly by introducing burst-limit policy. The ER history is used to infer a nodes new message transmission rate over time and the number of forwarded replicas per message. Nodes which send too many messages or replicas can thus be detected. Since ERs allows monitoring the sending behaviors of nodes over long time period, FDER could detect the burst traffic violation faster than the related work of Li.

FDER once more time confirms the benefits of using ERs for distributed monitoring and misbehavior detection in DTN. Since FDER and SDBG are both based on ERs, these two schemes could be integrated into a robust misbehavior detection system that can detect multiple attacks with little extra overhead.

The burst-limit policy creates a trade-off between flexibility and fairness. To address the trade-off, the author has designed a forwarding and queuing policy to lower the forwarding priority of messages that are generated by nodes with high rates of transmitting new messages. This policy is shown to ensure the delivery performance fairness among nodes with different traffic patterns. Besides, it provides a robust defense line against strategic attackers that continuously generate legitimate bursts to maximize the attack effects and hijack more network resources for themselves.

7.1.4 Security and privacy balance

In chapter 6, the author provides the balance between security and privacy in VANET by proposing a framework for vehicles to manage these two factors adaptively. ALRS supports adaptive privacy by providing three levels of recognition and selective recog-

dition among nodes despite the pseudonyms use. It is noted that a vehicle can enable or disable these linkability properties in a non-interactive way, without the involvement of other nodes. Therefore, vehicles can easily switch among the three recognition levels with certain nodes to adjust their privacy level flexibly. ATMS supports adaptive trust management in two aspects. First, a vehicle adapts the approach to evaluate event data, i.e. either objective or subjective, based on its distance to the data originator. Second, a vehicle adapts the entity trust setting based on the linkability level granted by other nodes in ALRS. Due to these adaptive factors, the proposed framework outperforms the related work SBAP in both trust and privacy management.

When ALRS and ATMS are used together, the trade-off between trust management and privacy among internal nodes still exists. The more privacy a node can achieve, the less information it provides other nodes for trust management and vice versa. However, this trade-off is less serious compared to the trade-off between the vehicles privacy against external eavesdroppers and trust management. Besides, the trade-off is mitigated by two factors. First, vehicles can adapt the privacy to the context, i.e. hiding identity or trust level when privacy is prioritized or revealing identity when trust management is required. Second, the recognition by trust level plays as a sweet spot to balance between trust management and privacy.

7.2 Future Research Directions

In the following, the author presents existing problems that the thesis has not solved as future research directions.

For detecting wormhole attack in DTN, the infrastructure requirement raises the issue of scalability in the larger area. The future work can focus on designing a more optimal layout and schedule for infrastructure nodes to further reduce the infrastructure

cost and to achieve acceptable detection time at the same time.

The detection schemes of blackhole, greyhole and flooding attacks in DTN are based on ERs and identity-based reputation systems. These factors prevent the nodes from hiding their identities and protecting the privacy. In DTN that involves the participation of users such as PSN, privacy might be considered a critical issue. Such scenario will require a new scheme to address both privacy and security against those attacks in DTN.

In the trust-privacy framework, the trust linkability provides a sweet spot to benefit both trust management and privacy protection. However, the trust recognition protocol in ALRS relies on the PSI-CA protocol which requires interactive communication and high communication overhead. Therefore, future work can target at designing a trust recognition protocol at lower cost and better scalability so that the trust linkability could be utilized more efficiently.

Bibliography

- [1] K. Fall, “A delay-tolerant network architecture for challenged internets” , in *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications, SIGCOMM*, pp. 27-34, 2003.
- [2] M. J. Khabbaz, C. M. Assi, and W. F. Fawaz. “Disruption-tolerant networking: A comprehensive survey on recent developments and persisting challenges”, in *IEEE Communications Surveys and Tutorials*, vol. 14, pp. 607-640, 2012.
- [3] Delay-Tolerant Networking Research Group, <http://www.dtnrg.org/wiki>.
- [4] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh and D. Rubenstein, “Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with-ZebraNet ”, in *International Conference on Architectural Support for Programming Languages and Operating Systems*, December 2002.
- [5] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot, “Pocket switched networks and human mobility in conference environments ”, in *Proceedings of the ACM SIGCOMM workshop on Delay-tolerant networking*, pp. 244-251, 2005.
- [6] J. Ott and D. Kutscher, “A Disconnection-Tolerant Transport for Drive-thru Internet Environments ”, in *Proc. IEEE Infocomm*, pp. 1849-1862, March 2005.
- [7] S. Guo, M. H. Falaki, E. A. Oliver, S. Ur Rahman, A. Seth, M. A. Zaharia and S. Keshav, “Very low-cost Internet access using KioskNet”, in *ACM SIGCOMM*, vol. 37, pp. 95-100, 2007.
- [8] A. Vahdat, and D. Becker, “Epidemic routing for partially-connected ad hoc networks,” Technical report, Duke University, 2000.
- [9] A. Lindgren, A. Doria, and O Schelen, “Probabilistic routing in intermittently connected networks”, in *SIGMOBILE Mobile Computing and Communication Review*, 2003.

- [10] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, "Maxprop: Routing for vehicle-based disruption-tolerant networks", in *Proceedings of IEEE Infocomm*, Barcelona, Spain, April 2006.
- [11] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and wait: an efficient routing scheme for intermittently connected mobile networks", in *Proceedings of the 2005 ACM SIGCOMM workshop on Delay tolerant networking*, pp.252-259, Philadelphia, USA, August 2005.
- [12] H. Hartenstein, "VANET: vehicular applications and inter-networking technologies ", in *Chichester, UK: Wiley*, vol. 1, 2010.
- [13] H. Hartenstein, and K. P. Laberteaux, "A tutorial survey on vehicular ad hoc networks ", in *IEEE Communications Magazine*, vol. 46, no. 6, pp. 164-171, 2008.
- [14] X. Yang, J. Liu, N. F. Vaidya, and F. Zhao, "A vehicle-to-vehicle communication protocol for cooperative collision warning", in *The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MOBIQ-UITOUS)*, pp. 114-123, 2004.
- [15] B. Ostermaier, F. Dtzer, and M. Strassberger, "Enhancing the security of local dangerwarnings in vanets-a simulative analysis of voting schemes ", in *The Second International Conference on Availability, Reliability and Security (ARES)*, IEEE, pp. 422-431, 2007.
- [16] Y. Zang, L. Stibor, HJ. Reumerman, and H. Chen, "Wireless local danger warning using inter-vehicle communications in highway scenarios", in *14th European Wireless Conference, (EW)*, pp. 1-7, 2008.
- [17] S. U. Rahman, and U. Hengartner, "Secure crash reporting in vehicular ad hoc networks", in *IEEE Security and Privacy in Communications Networks and the Workshops (SecureComm)*, pp. 443-452, 2007.
- [18] T. Nadeem, S. Dashtinezhad, C. Liao, and L. Iftode, "Trafficview: A scalable traffic monitoring system", in *Proceedings of IEEE International Conference on Mobile Data Management*, pp. 13-26, 2004.
- [19] R. Lu, X. Lin, H. Zhu, and X. Shen, "SPARK: a new VANET-based smart parking scheme for large parking lots", in *IEEE INFOCOM*, pp. 1413-1421, 2009.
- [20] T. W. Chim, S. M. Yiu, L. Hui, and V. Li, "VSPN: VANET-based secure and privacy-preserving navigation", in *IEEE Transactions on Computers*, vol. 63, no. 2, pp. 510-524, 2014.

- [21] Z. Li, C. Liu, and C. Chigan, "On secure VANET-based ad dissemination with pragmatic cost and effect control", in *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 1, pp. 124-135, 2013.
- [22] J. Loo, J. L. Mauri, and J. H. Ortiz, "Mobile Ad hoc networks: current status and future trends ", in *CRC Press*, 2011.
- [23] B. Wu, J. Chen, J. Wu and M. Cardei, "A survey of attacks and countermeasures in mobile ad hoc networks ", in *Wireless Network Security*, Springer, pp. 103-135, 2007.
- [24] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing ". Available online <http://www.ietf.org/rfc/rfc3561.txt>. July 2003.
- [25] D. Johnson, Y. Hu, and D. Maltz, "The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4 ". Available online <http://www.ietf.org/rfc/rfc4728.txt>. February 2007.
- [26] G. He, "Destination-Sequenced Distance Vector (DSDV) Protocol ". Technical report, Helsinki University of Technology, Finland.
- [27] NS-3. Available online <http://www.nsnam.org/>
- [28] ONE. Available online <https://www.netlab.tkk.fi/tutkimus/dtn/theone/>
- [29] A. Keranen, J. Ott, and T. Karkkainen, "The one simulator for dtn protocol evaluation", in *Proceedings of the 2nd International Conference on Simulation Tools and Techniques (Simutools)*, pp. 55:1-55:10, Rome, Italy, Mar. 2009.
- [30] Y. Hu, A. Perrig and D. Johnson, "Packet leashes: A defend against wormhole attacks in wireless mobile ad hoc networks ", in *Proceedings of IEEE INFOCOM*, 2003.
- [31] L. Hu and D. Evans, "Using directional antennas to prevent wormhole attacks ", in *Network and Distributed System Security Symposium*, 2004.
- [32] R. He, G. Ma, C. Wang and L. Fang, "Detecting and locating wormhole attacks in wireless sensor networks using beacon nodes ", in *World Academy of Science, Engineering and Technology*, 2009.
- [33] S. Capkun, L. Buttyan and J. Hubaux, "SECTOR: Secure tracking of node encounters in multi-hop wireless networks ", in *Proceedings of ACM Workshop on Security of Ad Hoc and Sensor Networks*, pp. 21-32, 2003.

- [34] H. Vu, A. Kulkarni, K. Sarac, and N. Mittal, "WORMEROS: A new framework for defending against wormhole attacks on wireless ad hoc networks ", in *Proceedings of Conference on Wireless Algorithms, Systems and Applications*, pp. 491-502, 2008.
- [35] J. Eriksson, S. V. Krishnamurthy, and M. Faloutsos, "TrueLink: A practical countermeasure to the wormhole attack in wireless networks ", in *Proceedings of IEEE International Conference on Network Protocols*, pp. 75-84, 2006.
- [36] L. Buttyan, L. Dora and I. Vajda, "Statistical wormhole detection in sensor networks ", in *Second European Workshop (ESAS)*, pp. 128-141, 2005.
- [37] S. Song and H. Wu and B. Choi, "Statistical wormhole detection for mobile sensor networks, ", in *International Conference on Ubiquitous and Future Networks*, pp. 322-327, 2012.
- [38] N. Song, L. Qian, and X. Li, "Wormhole attacks detection in wireless ad hoc networks: A statistical analysis approach ", in *Proceedings of IEEE International Parallel and Distributed Processing Symposium*, 2005.
- [39] Z. Zhao, B. Wei, X. Dong, L. Yao and F. Gao, "Detecting wormhole attacks in wireless sensor networks with statistical analysis ", in *Proceedings of IEEE International Conference on Information Engineering*, pp. 251-254, 2010.
- [40] W. Znaidi, M. Minier and J. Babau, "Detecting and locating wormhole attacks in wireless sensor networks using local neighbourhood information ", in *IEEE Personal Indoor and Mobile Radio Communication Symposium*, 2008.
- [41] R. Maheshwari, J. Gao and S. R. Das, "Detecting wormhole attacks in wireless networks using connectivity information ", in *Proceedings of IEEE INFOCOM*, 2007.
- [42] Y. Ren, M. C. Chuah, J. Yang and Y. Chen, "Detecting wormhole attacks in delay tolerant networks ", in *IEEE Wireless Communication*, vol. 17, no.5, 2010.
- [43] N. Asokan, K. Kostianen, P. Ginzboorg, J. Ott, and C. Luo, "Towards securing disruption-tolerant networking", Nokia Research Center, Tech. Rep. NRC-TR-2007-007.
- [44] R. Patra, S. Surana, and S. Nedeveschi, "Hierarchical identity based cryptography for end-to-end security in DTNs", in *Intelligent Computer Communication and Processing (ICCP)*, pp. 223-230, 2008.
- [45] S. Roy, and M. Chuah, "Secure data retrieval based on ciphertext policy attribute-based encryption (CP-ABE) system for the DTNs ", Technical Report, 2009.

- [46] Z. Jia, X. Lin, S.H. Tan, L. Li, and Y. Yang, "Public key distribution scheme for delay tolerant networks based on two-channel cryptography", in *Journal of Network and Computer Applications*, pp. 905-913, 2012.
- [47] M. Chuah, P. Yang, and J. Han, "A ferry-based intrusion detection scheme for sparsely connected ad hoc networks", in *Proc. Workshop on Security for Emerging Ubiquitous Computing*, 2007.
- [48] Y. Ren, M. Chuah, J. Yang, and Y. Chen. "MUTON: Detecting malicious nodes in disrupt-tolerant networks", in *IEEE Wireless Communications and Networking Conference (WCNC)*, 2010.
- [49] N. Li, and S. K. Das, "A trust-based framework for data forwarding in opportunistic networks", in *Elsevier Journal on Ad hoc Networks*, vol. 14, pp. 1497-1509, 2013.
- [50] Z. Gao, H. Zhu, S. Du, C. Xiao, and R. Lu, "PMDS: A probabilistic misbehavior detection scheme toward efficient trust establishment in Delay-tolerant networks", in *IEEE Transactions on Parallel and Distributed Systems*, January 2014.
- [51] F. Li, J. Wu, and A. Srinivasan, "Thwarting blackhole attacks in disrupt-tolerant networks using encounter tickets", in *Proc. INFOCOMM*, 2009.
- [52] Y. Guo, S. Schildt, and L. Wolf, "Detecting blackhole and greyhole attacks in vehicular delay tolerant networks", in *IEEE Conference on Communication Systems and Networks (COMSNETS)*, Jan 2013.
- [53] Q. Li, and G. Cao, "Mitigating routing misbehaviors in disruption tolerant networks", in *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, April 2012.
- [54] K. Park, and H. Lee, "On the effectiveness of route-based packet filtering for distributed DoS attack prevention in power-law Internets", in *Proc. ACM SIGCOMM*, Aug. 2001.
- [55] T. Anderson, T. Roscoe, and D. Wetherall, "Preventing Internet denial-of-service with capabilities", in *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 1, pp. 39-44, 2004.
- [56] X. Liu, X. Yang, and Y. Lu, "To filter or to authorize: network-layer DoS defense against multimillion-node botnets", in *Proc. of the ACM SIGCOMM conference on Data communication (SIGCOMM 08)*, pp. 195-206, 2008.

- [57] R. Mahajan, S. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling high bandwidth aggregates in the network ", in *Proc. ACM SIGCOMM*, 2002.
- [58] B. Raghavan, K. Vishwanath, S. Ramabhadran, K. Yocum, and A. Snoeren, "Cloud control with distributed rate limiting ", in *Proc. ACM SIGCOMM*, 2007.
- [59] P. Yi, Z. Dai, Y. Zhong, and S. Zhang "Resisting flooding attacks in ad hoc networks ", in *International Conference on Information Technology: Coding and Computing (ITCC)*, vol. 2, pp. 657-662, 2005.
- [60] S. K. Shandilya, and S. Sahu, "A trust based security scheme for RREQ flooding attack in MANET ", in *International journal of computer applications*, vol. 5, no. 12, 2010.
- [61] M. Patel, S. Sharma, and D. Sharan, "Detection and prevention of flooding attack using SVM ", in *International Conference on Communication Systems and Network Technologies (CSNT)*, pp. 533-537, 2013.
- [62] Y. Guo, S. Gordon, and S. Perreau, "A flow based detection mechanism against flooding attacks in mobile ad hoc networks ", in *IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 3105-3110, 2007.
- [63] G. Ansa, H. Criuckshank, Z. Sun, and M. Al-Siyabi, "A DOS-resilient design for delay tolerant networks", in *7th IEEE International Wireless Communications and Mobile Computing Conference (IWCMC)*, pp. 424-429, 2011.
- [64] F. C . Lee, W. Goh, and C. K. Yeo, "A queuing mechanism to alleviate flooding attacks in probabilistic delay tolerant networks ", in *6th Advanced International Conference on Telecommunications (AICT)*, pp. 329-334. IEEE, 2010.
- [65] V. Natarajan, Y. Yang, and S. Zhu, "Resource-misuse attack detection in delay-tolerant networks ", in *IEEE 30th International Performance Computing and Communications Conference (IPCCC)*, pp. 1-8, 2011.
- [66] Q. Li, W. Gao, S. Zhu, and G. Cao, "To lie or to comply: defending against flood attacks in disruption tolerant networks ", in *IEEE Transactions on Dependable and Secure Computing*, vol. 10, no. 3, pp. 168-182, 2013.
- [67] F. Li, J. Wu, and A. Srinivasan, "Thwarting blackhole attacks in disrupt-tolerant networks using encounter tickets", in *Proc. INFOCOMM*, 2009.
- [68] S. Zhao, A. Aggarwal, R. Frost and X. Bai, "A survey of applications of identity-based cryptography in mobile ad-hoc networks ", in *IEEE Communications surveys & tutorials*, vol. 14, no.2, 2010.

- [69] L. Law, and B. J. Matt, "Finding invalid signatures in pairing-based batches ", in *Cryptography and Coding*, Springer Berlin Heidelberg, vol. 4887, pp. 34-53, 2007.
- [70] R. K. Schmidt, T. Leinmller, E. Schoch, A. Held, and G. Schfer "Vehicle behavior analysis to enhance security in vanets ", in *Proceedings of the 4th IEEE Vehicle-to-Vehicle Communications Workshop (V2VCOM2008)*, 2008.
- [71] N. Bimeyer, C. Stresing, and K. M. Bayarou, "Intrusion detection in vanets through verification of vehicle movement data ", in *IEEE Vehicular Networking Conference (VNC)*, pp. 166-173, 2010.
- [72] H. Stbing, A. Jaeger, N. Bimeyer, C. Schmidt, and S. A. Huss "Verifying mobility data under privacy considerations in Car-To-X communication ", in *Proceedings of 17th ITS World Congress*, 2010.
- [73] M. Ghosh, A. Varghese, A. A. Kherani, and A. Gupta, "Distributed misbehavior detection in VANETs ", in *IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1-6, 2009.
- [74] S. K. Harit, G. Singh, and N. Tyagi, "Fox-Hole Model for Data-centric Misbehaviour Detection in VANETs ", in *Third International Conference on Computer and Communication Technology (ICCCT)*, pp. 271-277, 2012.
- [75] S. Ruj, M. A. Cavenaghi, Z. Huang, A. Nayak, and I. Stojmenovic "On data-centric misbehavior detection in VANETs ", in *IEEE Vehicular Technology Conference (VTC Fall)*, pp. 1-5, 2011.
- [76] M. Gerlach, "Trust for vehicular applications", in *IEEE Eighth International Symposium on Autonomous Decentralized Systems (ISADS)*, pp. 295-304, 2007.
- [77] D. Huang, Z. Zhou, X. Hong, and M. Gerla, "Establishing email-based social network trust for vehicular networks", in *IEEE 7th Consumer Communications and Networking Conference (CCNC)*, pp. 1-5, 2010.
- [78] U. F. Minhas, J. Zhang, T. Tran, and R. Cohen, "Towards expanded trust management for agents in vehicular ad-hoc networks ", 2010.
- [79] M. Raya, P. Papadimitratos, V. D. Gligor, and J. P. Hubaux, "On data-centric trust establishment in ephemeral ad hoc networks ", in *the 27th Conference on Computer Communications (INFOCOM)*, IEEE, 2008.
- [80] Z. Huang, S. Ruj, M. A. Cavenaghi, M. Stojmenovic, and A. Nayak, "A social network approach to trust management in VANETs", in *Peer-to-Peer Networking and Applications*, vol. 7, no. 3, pp. 229-242, 2014.

- [81] F. Kargl, P. Papadimitratos, L. Buttyan, M. Muter, E. Schoch, B. Wiedersheim, and V. T. Ta, "Secure vehicular communication systems: implementation, performance, and research challenges", in *IEEE Communications Magazine*, vol.46, no. 11, pp. 110-118, 2008.
- [82] K. Zeng "Pseudonymous PKI for ubiquitous computing", in *Public Key Infrastructure*, Springer Berlin Heidelberg, pp. 207-222, 2006.
- [83] G. Calandriello, P. Papadimitratos, J.-P. Hubaux, and A. Liy, "Efficient and robust pseudonymous authentication in VANET ", in *Proc. 4th ACM Int. workshop VANET*, pp. 1928, 2007.
- [84] P. Kamat, A. Baliga, and W. Trappe "Secure, pseudonymous, and auditable communication in vehicular ad hoc networks ", in *Security and Communication Networks*, vol. 1, no. 3, pp.233-244, 2008.
- [85] B. H. Kim, K. Y. Choi, J. H. Lee, and D. H. Lee, "Anonymous and traceable communication using tamper-proof device for vehicular ad hoc networks ", in *IEEE International Conference on Convergence Information Technology*, , pp. 681-686, 2007.
- [86] X. Lin, X. Sun, P.-H. Ho, and X. Shen, "GSIS: A secure and privacy-preserving protocol for vehicular communications ", in *IEEE Transaction on Vehicular Technology*, vol. 56, no. 6, pp. 3442-3456, 2007.
- [87] L. Zhang, Q. Wu, A. Solanas, and J. Domingo-Ferrer, "A scalable robust authentication protocol for secure vehicular communications ", in *IEEE Transactions on Vehicular Technology*, vol. 59, no. 4, pp. 1606-1617, 2010.
- [88] Y. Sun, Z. Feng, Q. Hu, and J. Su, "An efficient distributed key management scheme for group-signature based anonymous authentication in vanet ", in *Security and Communication Networks*, vol. 5, no. 1, pp. 79-86, 2012.
- [89] D. Eckhoff, C. Sommer, T. Gansen, R. German, and F. Dressler, "Strong and affordable location privacy in VANETs: Identity diffusion using time-slots and swapping", in *Proc. 2nd IEEE VNC*, pp. 174181, Dec. 2010.
- [90] Y. Pan, J. Li, L. Feng, and B. Xu, "An analytical model for random changing pseudonyms scheme in VANETs ", in *Proc. Int. Conf. NCIS*, pp. 141145, 2011.
- [91] R. Lu, X. Li, H. Luan, X. Liang, and X. Shen. "Pseudonym changing at social spots: An effective strategy for location privacy in vanets ", in *IEEE Transactions on Vehicular Technology*, vol. 61, no. 1, pp. 86-96, 2012.

- [92] B. Chaurasia and S. Verma, "Optimizing pseudonym updation for anonymity in VANETs", in *Proc. IEEE APSCC*, pp. 1633-1637, Dec. 2008.
- [93] K. Sampigethaya, M. Li, L. Huang, and R. Poovendran "AMOEBA: Robust location privacy scheme for VANET ", in *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 8, pp. 1569-1589, 2007.
- [94] Q. Wu, J. Domingo-Ferrer, and U. Gonzlez-Nicols "Balanced trustworthiness, safety, and privacy in vehicle-to-vehicle communications", in *IEEE Transactions on Vehicular Technology*, vol. 59, no. 2, pp. 559-573, 2010.
- [95] L. Chen, Q. Li, K. Martin, and S. Ng, "A privacy-aware reputation-based announcement scheme for VANETs ", in *WiVeC*, pp. 1-5, 2013.
- [96] Y. Wei, Y. Chen, and H. Shan "Beacon-based trust management for location privacy enhancement VANETs ", in *the 13th Asia-Pacific Network Operations and Management Symposium (APNOMS), IEEE, 2011* , pp. 1-8, 2011.
- [97] M. J. Freedman, K. Nissim, and B. Pinkas, "Efficient private matching and set intersection ", in *Advances in Cryptology-EUROCRYPT*, Springer Berlin Heidelberg, pp. 1-19 , 2004.
- [98] L. Kissner, and D. Song, "Privacy-preserving set operations ", in *Advances in CryptologyCRYPTO 2005*, Springer Berlin Heidelberg, pp. 241-257, 2005.
- [99] E. De Cristofaro, and G. Tsudik "Practical Private Set Intersection Protocols with Linear Computational and Bandwidth Complexity ", in *IACR Cryptology ePrint Archive*, 2009.
- [100] S. Jarecki, and X. Liu "Fast secure computation of set intersection ", in *Security and Cryptography for Networks*, Springer Berlin Heidelberg, pp. 418-435, 2010.
- [101] A. Yao "Protocols for secure computations ", in *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pp. 160-164. IEEE, 1982.
- [102] D. Many, M. Burkhart, and X. Dimitropoulos, "Fast private set operations with sepia ", in *Technical Report 345*, Mar, 2012.
- [103] F. Kerschbaum, "Outsourced private set intersection using homomorphic encryption ", in *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, pp. 85-86, 2012.
- [104] M. Lentz, V. Erdlyi, P. Aditya, E. Shi, P. Druschel, and B. Bhattacharjee, "SDDR: light-weight, secure mobile encounters ", in *23rd USENIX Security Symposium*, pp. 925-940, 2014.

- [105] B. H. Bloom “Space/time trade-offs in hash coding with allowable errors”, in *Communications of the ACM*, no. 7, pp. 422-426, 1970.
- [106] H. Lim, N. Lee, J. Lee, and C. Yim, “Reducing false positives of a Bloom filter using cross-checking Bloom filters ”, in *Appl. Math*, vol.8, no.4, pp.1865-1877, 2014.
- [107] M. Freedman, K. Nissim, and B. Pinkas, “Efficient private matching and set intersection ”, in *Eurocrypt*, 2004.
- [108] S. Hohenberger and S. Weis, “Honest-verifier private disjointness testing without random oracles”, in *PET*, 2006.
- [109] L. Kissner and D. Song, “Privacy-preserving set operations ”, in *Crypto*, 2005.
- [110] J. Vaidya and C. Clifton, “Secure set intersection cardinality with application to association rule mining ”, in *Journal of Computer Security*, vol. 13, no. 4, 2005.
- [111] E. De Cristofaro, P. Gasti, and G. Tsudik, “Fast and private computation of cardinality of set intersection and union ”, in *Cryptology and Network Security*, Springer Berlin Heidelberg, pp. 218-231, 2012.

Author's Publications

Journals

- **Thi Ngoc Diep Pham** and Chai Kiat Yeo, "Adaptive Trust and Privacy Framework for Vehicular Networks ", submitted to *IEEE Transactions on Vehicular Technology*.
- **Thi Ngoc Diep Pham** and Chai Kiat Yeo, "Detecting Flooding Attack in Delay Tolerant Networks by Piggybacking Encounter Records ", submitted to *IEEE Transactions on Vehicular Technology*.
- **Thi Ngoc Diep Pham** and Chai Kiat Yeo, "Detecting Colluding Blackhole and Greyhole Attacks in Delay Tolerant Networks ", *IEEE Transactions on Mobile Computing*, 15(5), 1116-1129, 2016.

Conferences

- **Thi Ngoc Diep Pham** and Chai Kiat Yeo, "A Trust-Privacy Framework for Vehicular Networks ", in *Proceedings of Wireless Telecommunication Symposium (WTS)*, London, England, Apr 2016.
- **Thi Ngoc Diep Pham** and Chai Kiat Yeo, "Detecting Flooding Attack in Delay Tolerant Networks by Piggybacking Encounter Records ", in *Proceedings of International Conference on Information Science and Security (ICISS)*, Seoul, Korea, Dec 2015.
- **Thi Ngoc Diep Pham** and Chai Kiat Yeo, "Detecting Colluding Blackhole and Greyhole Attacks in Delay Tolerant Networks ", in *Proceedings of IEEE Consumer Communications and Networking Conference (CCNC)*, Las Vegas, Nevada, pp. 233-238, Jan 2015.
- **Thi Ngoc Diep Pham** and Chai Kiat Yeo, "Statistical Wormhole Detection and Localization in Delay Tolerant Networks", in *Proceedings of IEEE Consumer Communications and Networking Conference (CCNC)*, Las Vegas, Nevada, pp. 380-385, Jan 2014.