Intrusion detection and tolerance for wireless mesh networks.

Liu, Peng

Intrusion Detection and Tolerance for Wireless Mesh Networks

Peng LIU

573466

Submitted to Swansea University in fulfilment of the requirements for the Degree of Master of Philosophy

Swansea University
Prifysgol Abertawe

College of Engineering

Swansea University

Under Supervision of Dr. Xinheng Wang

January 2012

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

ABSTRACT

When most of the research efforts have been placed on the applications of Wireless Mesh Networks (WMNs), their security has increasingly raised academic concerns. Because there are always some new attacks having immunity against the existing prevention mechanisms, intrusion detection as a second defence has been proposed to mitigate the effects of malicious network intruders.

The network security is categorised into data availability, confidentiality, authenticity and integrity. Although cryptography mechanisms have protected the last three security services to some degree, the attacks against the availability are not straightforward to handle because it is difficult to distinguish whether service disruption of a node results from random link failures or attacks.

To protect the availability of WMNs, this thesis proposes a secure Ad-hoc On-demand Distance Vector (AODV) routing protocol as an intrusion detection scheme to minimize the effects of attacks, especially for the packets drop attack.

Specifically, the reputation mechanism is applied to address the issue between malicious attacks and random failures. To compute a consensus reputation for each node, the EigenTrust algorithm is utilized in AODV. Because the convergence of the algorithm is guaranteed by the pre-defined trusted nodes but in AODV such nodes are not defined, the algorithm is improved in terms of modifying initial value and adding a subsidiary termination condition to warrant the convergence. Besides, the application of EigenTrust merges the data fusion and reputation calculation layers in traditional functional framework of intrusion detection.

Moreover, AODV is modified to work with EigenTrust. The Route Request (RREQ) message is attached extensions including reputation information and the procedure of processing routing messages is also correspondingly altered.

Furthermore, the improved AODV based on EigenTrust has been implemented in Linux. Experimental results show that it is able to select a relative reliable route excluding the node compromised by the packet drop attack.

DECLARATION

This work has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed . ..... (candidate)

Date 08/03/2012

STATEMENT 1

This thesis is the result of my own investigations, except where otherwise stated. Where correction services have been used, the extent and nature of the correction is clearly marked in a footnote(s).

Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed .. .. (candidate)

Date 08/03/2012

STATEMENT 2

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed .. .. (candidate)

Date 08/03/2012

PUBLICATIONS

A Secure AODV Protocol Based on Distributed EigenTrust Algorithm against Packet Drop Attacks in WMNs, Submitted to *Security in Wireless Ad Hoc and Sensor Networks with Advanced QoS Provisioning* on Wiley Security and Communication Networks Journal.

TABLE OF CONTENTS

ACKNOWLEDGEMENT

It is my pleasure to thank those who help and support me throughout the research and thesis writing period.

Firstly, I am heartily thankful to my supervisor, Dr. Xinheng Wang, whose patience, encouragement, supervision and support from the preliminary to the concluding level enabled me to develop an understanding of the subject. I could not have imagined having a better supervisor for my M.Phil. study.

Secondly, I would like to thank Professor Thomas Chen who is always willing to help me and give his best advice. My research would not have been possible without his advice.

Thirdly, I also would like to express my sincere gratitude to Dr. Shancang Li who always gives me the valuable technique assistance, enlightens me in the research and helps me in daily life.

Besides, I extremely cherish the time with my housemates: Haitao Yang and Avril Cui whom I have wonderful time with. I also thank my friends and lab mates in room 003 in Technium Digital Building of Swansea University.

Another person I would like to appreciate is Mr. Jerome Bradnick who spent his personal time on checking my thesis and gives me advice on vocabulary and grammar. I really appreciate him from the bottom of my heart.

Finally, my best regards and blessings must also be given to my parents and fiancée who are always supporting me during my study.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AODV	Ad-hoc On-demand Distance Vector
AODV-REX	AODV-Reputation Extension
AODV-ET	AODV-EigenTrust
AODV-UU	AODV-Uppsala University
AP	Access Point
API	Application Programming Interface
BPF	Berkeley Packet Filter
GCHQ	Government Communication Head Quarters
CONFIDANT	Cooperation Of Nodes: Fairness In Dynamic Ad-hoc NeTworks
CORE	COllaborative Reputation
D-S	Dempster-Shafer
DSR	Dynamic Source Routing
GCC	GNU Compiler Collection
HIDS	Host-based Intrusion Detection System
IDS	Intrusion Detection System
IP	Internet Protocol
LAN	Local Area Network
MAC	Media Access Control
MANET	Mobile Ad-hoc Networks
NIC	Network Interface Card
ND	Network Diameter
NIDS	Network-based Intrusion Detection System
REFACING	RElationship FAmiliarity Confidence INteGrity
RREP	Route REPly
RREQ	Route REQuest
RSA	Rivest, Shamir and Adleman
RT	Reputation table
SOPE	Self-Organized Protocol for Evaluating
SORI	Secure and Objective Reputation-based Incentive
SPF	Shortest Path First
SVM	Support Vector Machine
WLAN	Wireless Local Area Network
WMN	Wireless Mesh Network

1 Introduction

1.1 Background

Nowadays wireless communication technologies offer people an efficient and flexible way of communicating, which transfer information between two or more points through air instead of cables.

A Wireless Local Area Network (WLAN) is a kind of such technology, which enables users to access a Local Area Network (LAN) through the radio connections rather than traditional network cables. It extends an existing backbone wired LAN in large areas, for example in manufactory plants and office buildings. As shown in Figure 1-1, WLAN is built by connecting fixed access points to the backbone wired network. Each terminal user communicates with an Access Point (AP) using a wireless network adapter similar in function to a traditional Ethernet adapter.

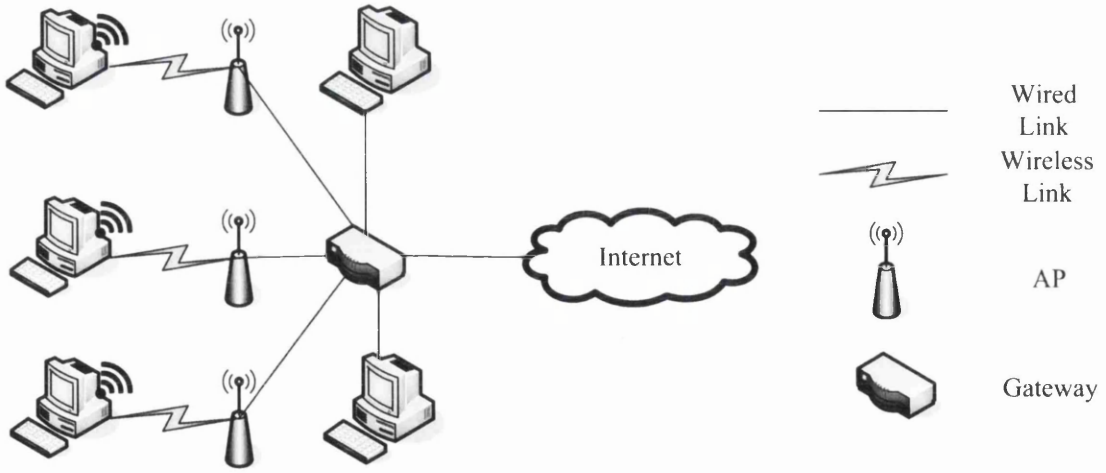

Figure 1-1 Architecture of WLAN

However, because wireless radio transmission signal attenuates with distance significantly, the short radio range hinders the application of WLAN in a large area[1]. Besides, the mobility between APs is also limited so that some services like VoIP (Voice over IP) run unsmoothly[2].

The appearance of a self-organized wireless networking technology named Wireless

Mesh Network (WMN) has broken through those limitations and promoted the growth of WLAN. Currently, increasing deployments of WMNs are scattered around us. For example, community and neighbourhood networks where mesh routers are placed on the roof of houses in a neighbourhood, which serve as APs for users inside the houses and along the roads[3]. Another common example in our daily life is that a mobile phone accesses Internet service via Wi-Fi signal supplied by another one that is set to hotspot mode and functions as an AP.

However, the WMN also confronts increasing threats on its security, like deliberate disruption of service of a network (availability). Corresponding countermeasures have been proposed to defend the network, such as deploying firewall, applying encryption mechanism, establishing authentication systems, and configuring wireless devices.

Different from those methods, a relative new technology named Intrusion Detection System (IDS) has been deployed as a line of passive defence, like a burglar alarm in a building. The IDS determines possible attacks on a network by auditing and analyzing information from a variety of sources: network interfaces, host log files, etc.

1.2 Motivation for Security

Security issues cannot be ignored for WMNs when they are deployed, because such networks are vulnerable to a variety of threats due to its dynamically changing topology, absence of conventional security infrastructures as well as open communication medium. So, in operation, relevant security measures must be considered.

1.2.1 The Need for Defence

A widely used method for warranting the security of WMN is by deploying a firewall to prevent attacks, but, learnt from experience, there are always some new attacks having immunity against the existing prevention method. It is necessary to deploy a kind of system that is capable of detecting potential attackers who have intruded into the network. Such a type of system is called Intrusion Detection System.

Another countermeasure is to use a cryptography method that can transform a piece of message into unreadable cipher text. Only the intended recipient keeping a secret key can decipher the text to read. It was generally accepted that cryptographic methods could well guarantee the security of data over wireless networks. Currently, however, encrypted messages can be cracked, especially with the assistance of specific software and powerful computing devices, although modern cryptography techniques were believed to be unbreakable. In addition, once leakage of the secret key occurs, a succeeded intruder could easily launch attacks to compromise a network. More importantly, encryption measures can only offer protection to a network in terms of information confidentiality, authenticity and integrity, except for availability. But intrusion detection is able to judge whether information and services are available by monitoring network activities.

In short, there is a strong demand on the intrusion detection scheme that is a complementation to existing security mechanisms, for resisting against the achieved attacker in WMNs.

1.2.2 The Need for Secure Routing Protocol

One common type of threats to WMNs is the attacks against the routing protocols that utilize routing messages to establish routes to destination nodes and transfer data packets along the established routes to the destinations.

Attacks on routing messages, such as Message Modification, Man-in-the-Middle etc., can be avoided by using encryption mechanism to some degree. But it is rather difficult to deal with the attacks against a routing behaviour of data packets. In general, if a compromised node has been authenticated to participate in routing activities, other nodes cannot distinguish whether a legal routing message is coming from normal nodes or the malicious node that actually denies propagating data packets. Such misbehaviour causes negative effect on the availability of data over the network.

To mitigate impact of the misbehaviour, an indicator of data routing behaviour should be placed in routing protocols for WMNs. This indicator could be a reputation

metric representing the trustworthiness and the data transferability of each node in the network in order to avoid the impact of compromised nodes.

1.2.3 The Need for Reputation Mechanism

Due to distributed and self-organized nature of WMN, the IDS for WMN should depend on cooperation among nodes. That is, a node makes a decision based on its co-operators' opinions, instead of acting arbitrarily in its own view. Obviously, a node with occasional link break should be more prone to be a data relay node, compared with a compromised node. But in reality it is difficult to determine that the dropping packets are caused by occasional link failure or a deliberate attack on the availability.

Reputation mechanism can be used to address this problem to some extent. In a reputation mechanism, each node keeps track of its neighbourhood's behaviour and exchanges the records with them in order to calculate a comprehensive reputation value. Based on this reputation, the computing node will cooperate with the nodes that enjoy high reputation values later. For example, a node can select a reliable route including trustful nodes to transmit data packets rather than the route forwarding through the nodes with low reputations. However, because methods of reputation calculation in many reputation schemes do not combine opinions from multiple nodes, different nodes generally hold various evaluations about one's reputation. To address this issue, this paper employs a distributed EigenTrust[4] algorithm to compute a comprehensive reputation for each node in WMNs. Necessary modifications to the application of the algorithm is needed.

In summary, this thesis is attempting to address the issue of availability loss (packet dropping) resulting from an intruder who has successfully compromised a partial amount of nodes in a WMN. In particular, a node is able to select the trustworthy partners to accomplish data transfer, meanwhile avoid those untrusted. Additionally, this kind of method also enhances the tolerance to the compromised nodes in the WMN.

1.3 Contributions

The major contribution of this thesis is developing a reputation based routing protocol for intrusion detection to mitigate the effects of the attacks on the data availability of WMNs.

Specifically, the distributed EigenTrust[4] algorithm is improved in Chapter 4 to calculate the reputation of every node in a WMN. The convergence of the algorithm can be guaranteed in any situation when trusted nodes are pre-defined. But, most routing protocols for WMNs do not assume the pre-trusted nodes. Therefore, the algorithm is modified in terms of initial value and termination condition to ensure that the convergence of the algorithm would not be compromised when no pre-trusted nodes are specified.

Moreover, the functional framework of intrusion detection proposed in [5] is simplified and applied into practice. Section 3.4 merges the data fusion and reputation calculation layers in the functional framework of intrusion detection described in Section 3.2.2. This is because the EigenTrust algorithm can figure out one node's reputation based on synthesising reputations of co-operators. The application of the algorithm can also eliminate the interference due to temporary or occasional link fault.

Furthermore, based on the improved EigenTrust algorithm and the simplified functional framework, a secure AODV[6] routing protocol is designed in Chapter 5 as an intrusion detection scheme for WMNs. The proposed secure AODV does not change any field size in the routing messages of AODV but attaches reputation extensions to the standard route request message of AODV in order to implement EigenTrust into AODV. Every extension includes trustworthiness of a rater and a ratee as well as their identities. Each node calculates its own reputation based on the data in its received extensions. Also, the procedure of processing routing messages is correspondingly altered slightly.

Finally, the secure AODV is developed and implemented in Linux operating system. Details in the implementation are also illustrated and experimental results show that the implementation can keep away from abnormal nodes having been compromised

by the packet drop attack.

Overall, this thesis makes contributions to the security of the data availability of WMNs. An EigenTrust based AODV routing protocol for intrusion detection, co-operatively computing the comprehensive reputation for every node in WMNs, is implemented to mitigate the availability loss resulting from the packets drop attack.

1.4 Scope and Limitation

It is should be pointed out that the implementation in this thesis is not a versatile solution to all security issues of WMNs.

First, this thesis does only concentrate on mitigating the impact of the packet drop attack that causes the data availability loss in WMNs. Of course, other attacks on availability, e.g., Rushing attack and Wormhole attack, can also be resisted to some extent, since these attacks both aim to drop packets.

Second, the implementation does not supply protection against attacks on confidentiality, authenticity and integrity of WMNs. Because cryptography has been used to satisfy those three security requirements, such as public-key and digital signature. Of course, existing encryption mechanisms can be employed to strengthen our intrusion detection scheme.

1.5 Thesis Organization

Chapter 2 generalizes the challenges in the WMN and then concentrates on the security attacks faced by the WMN. At last, it presents the corresponding countermeasures to the threats against confidentiality, authenticity and integrity, and introduces the intrusion detection to overcome the attack on availability.

Chapter 3 describes the conventional system architectures and approaches of intrusion detection. Then it reviews the recent research about the distributed, cooperative and reputation based intrusion detection schemes that are the basis of our intrusion detection system.

Chapter 4 describes and stimulates a reputation algorithm named EigenTrust. After

that, it brings the algorithm into our intrusion detection scheme. Furthermore, some issues in the application of the algorithm are addressed.

Chapter 5 illustrates an improved AODV routing protocol based on the EigenTrust algorithm. Besides, a testbed is set up and the experimental procedure is depicted. The experimental results show that the improved AODV is able to select a relative trustworthy path to avoid the node compromised by the packet drop attack.

Chapter 6 concludes this thesis and gives out the direction of further work.

2 Wireless Mesh Network

The wireless mesh network is different from a traditional wired network in terms of architecture and transmission media. The differences result in that the existing solutions, e.g., routing protocols and security mechanisms, for the wired networks are not necessarily valid for the WMNs. To support the feasibility of the implementation of intrusion detection developed in the following chapters, the unique characteristics and the security challenges faced by WMNs are enunciated in this chapter.

2.1 Architecture

As shown in Figure 2-1, the WMN is often composed of mesh clients, APs and one or more gateways. The clients are often laptops, mobile phones and other wireless devices while the mesh APs, which extend the WLAN and form the mesh network, transfer network traffic to and from the gateway that may, but not necessary, connect to the Internet. Unlike WLAN where an AP only transfers data to the end users within its transmission range and communication between the users must be completed via APs, every node (i.e., APs and clients) in the WMN collaborates to propagate data and both clients and APs can directly transfer data to the intended receiver in their coverage without the assistance of the third party.

The WMN is a mix of the fixed and mobile wireless devices. It is regarded as one type of wireless ad-hoc network where each node is capable of participating communication without the support of pre-deployed infrastructures like APs in the WLANs and routers in the wired networks. Therefore, the existing approaches, such as routing protocols, proposed under the background of ad-hoc networks[7] are also valid for WMNs.

Due to their popularity and vulnerability to the attacks, the WMN is drawing the attention of network attackers. For instance, the routing protocol generally becomes a victim of attacks that decrease the network performance.

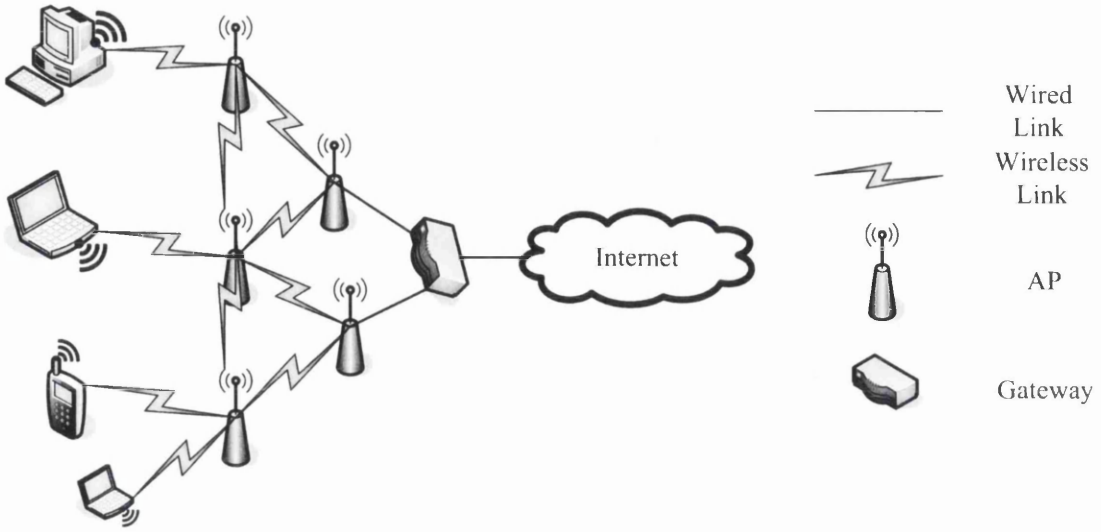

Figure 2-1 Architecture of WMN

2.2 Characteristics and Challenges

The wireless mesh network possesses different characteristics from the traditional wired network and also faces the inherent challenges in ad-hoc networks.

Firstly, the wireless medium is an open, shared, and broadcast medium. Any person, including an attacker, can capture and analyse the traffic by monitoring the radio medium in the WMN. This leaves a door open for malicious attackers.

Another special nature of WMN is self-organisation. A mesh node may be mobile so that the topology of WMN sometimes changes. In addition, wireless connections may fail occasionally, due to signal noise, channel interference, traffic congestion and so on. So the WMN is capable of self-management and self-healing.

Moreover, radio path loss is severer in the wireless environment than in the wired. It is widely accepted that wireless radio transmission signal attenuates with distance. According to the classical physics, the attenuation of radio signal is proportional to the inverse square of distance. Moreover, experimental evidence shows that it is actually far greater than that, especially for indoor environments. Consequently, nodes in a wireless network probably have relatively limited coverage, which results in acquiring limited information only about local range.

Furthermore, the short transmission range leads to that wireless signal must be

relayed by intermediate nodes if a sender node intends to communicate with a remote receiver. But available channel capacity to users in each AP declines rapidly as more intermediate APs are added[8]. Thus, the routing protocols for WMNs should not only offer multi-hop routing but also optimize routing behaviour to reduce unnecessary transmissions overhead.

Therefore, the above discussed challenges deserve careful consideration in the application of WMN technologies. Specific protocols for ad-hoc networks have been proposed to endue mesh nodes autonomous, multi-hop transmission and to optimise the transmission overhead. The next section describes a particular routing protocol for ad-hoc networks, and following that, the security challenge of WMN are illustrated.

2.3 Routing Protocols for Wireless Mesh Networks

A routing protocol for a WMN is responsible for deciding behaviour of nodes that are not familiar with the topology of the network. In general, the routing protocol for WMN can be divided into two types: proactive (table driven) and reactive (on demand) protocols. Even though the routing protocols are originally designed for ad-hoc networks, they also can be used in WMNs.

2.3.1 Proactive Routing Protocol

Proactive routing protocols maintain routes to all destinations as a form of routing table, regardless of whether or not these routes are needed. A node must periodically send routing messages in order to maintain up-to-date route information. Therefore, proactive routing protocols generate much transmission overhead, since routing messages are still sent out when there is no data packet, even though a node can quickly obtain route information to an intended destination compared with reactive routing protocols.

2.3.2 Reactive Routing Protocol

Reactive routing protocols do not maintain routing information to all nodes. Each node only stores routing information about its relay node and destination node. A

node only searches or updates a route to a destination when it has a need to send or forward data packets. So, reactive routing protocols do not consume unnecessary routing overhead to maintain its routing table although they probably spend more time on route finding, compared against proactive protocols.

1) AODV Routing Protocol

AODV is a typical reactive routing protocol. It establishes a route to a destination only on demand by transmitting and receiving routing messages such as RREQ, RREP, RERR, and HELLO.

(1) Route Discovery

Figure 2-2 assumes that each node merely knows who its neighbours are using HELLO messages explained later. Node S is attempting to send data to the destination D but it has not established a route to D. So, it generates and broadcasts a Route REQUEST (RREQ) message to ask its neighbours (i.e., A and B) for a route to D. When the neighbours received such a RREQ, they separately increase the Hop Count field of the RREQ by 1 and forwarded the message to their own neighbours because so far they have not known any information about the destination yet. It should be noted that the source node S will also receive its own RREQ back from A and B. In this situation, S just ignores and discards this message. Similarly, node A does not also broadcast the same RREQ which it has forwarded before.

If the propagation of the RREQ goes continually, at last the RREQ message will arrive at the expected destination (D) with help of intermediate nodes (e.g., node B). Once D recognises that its IP address is identical with the field of Destination IP Address included in the RREQ, it will launch a route reply for responding to the RREQ.

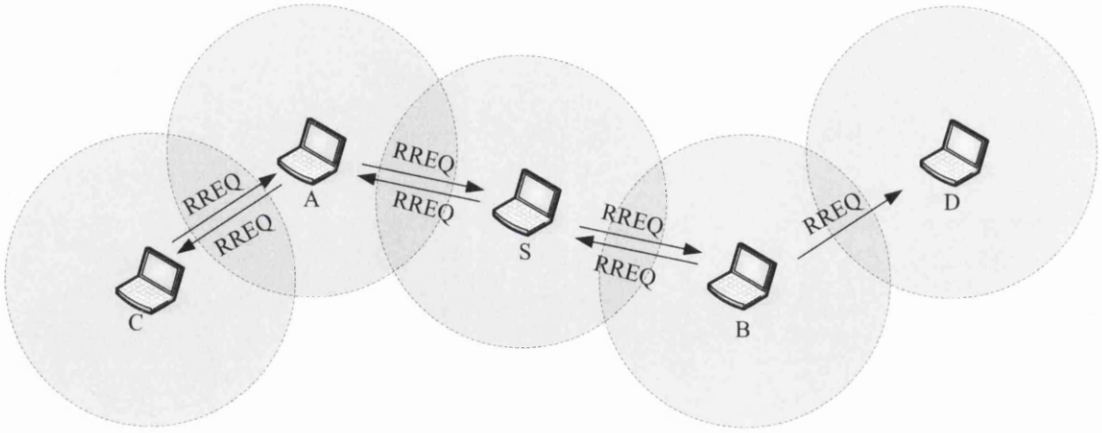

Figure 2-2 Route Discovery

(2) Route Response

Different from the broadcasted RREQ, a Route REPLY (RREP) is a unicasted routing message back to the originator of a corresponding RREQ. If an intermediate node has a route to a destination during the propagation of the RREQ, it will reply the RREQ on behalf of the destination. Otherwise, like in Figure 2-3, only the destination D itself replies the RREQ by unicasting a RREP back to the upstream node B.

It probably happens in AODV that a node receives many same RREQs routed by different neighbours. To minimize unnecessary routing overhead, nodes only reply the first coming RREQ with the same sequence number. The sequence number has similar function as time stamp, identifying a message and indicating how fresh it is. Every time a node generates any type of routing message, it will increase its own sequence number counter by 1.

While the RREP is forwarded back toward the RREQ source in Figure 2-3, its Hop Count field is incremented by 1 at each node as well. Thus, when S receives the RREP, it can know the distance in hops to the destination by the Hop Count. If S does not receive a reply within a specified time, it will rebroadcast a new RREQ with a larger sequence number.

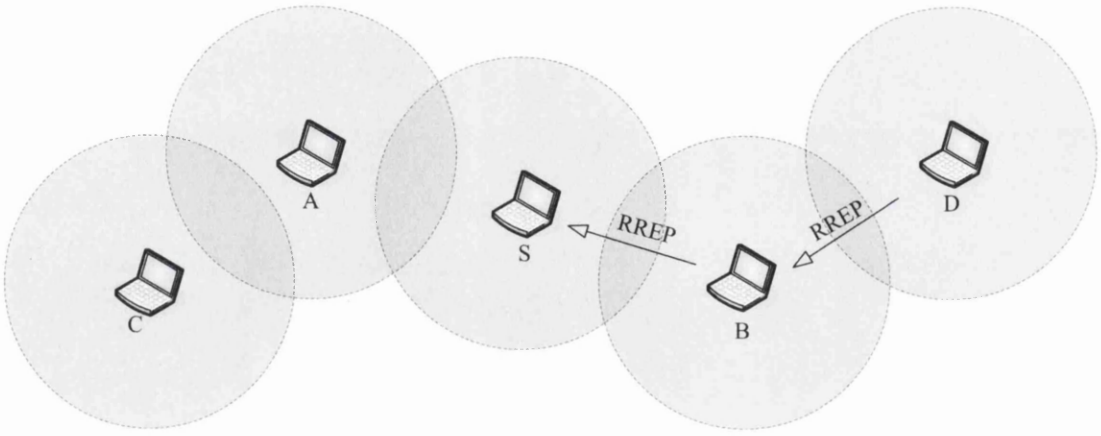

Figure 2-3 Route Reply

(3) Route Selection

Shortest Path First (SPF) is often widely used in network routing protocols (no exception to AODV) to select a route. For example, if a RREQ source node receives several RREPs with the same sequence number coming from various paths, it will find the RREP with the least hops, and then the neighbour that forwarded the RREP back will be selected as a relay to forward data packets.

(4) Neighbours Management

Nodes learn of their neighbours through two ways. In phrases of start-up or no data packets to send, a node keeps acquaintance with its neighbours by broadcasting a HELLO message, a special RREP message, which is only allowed to spread over one hop.

The other way is through the dissemination of the RREQ. Whenever a node receives a RREQ from a neighbour, it updates its local connectivity entries about this neighbour.

(5) Route Maintenance

Every node maintains two types of routing tables storing route information, namely a reverse routing table and a forward routing table. The reverse route is set up for unicasting the RREP back to a source. While the RREP is unicasted by each intermediate node, a forward route heading to a destination is established as well.

When receiving a broadcasted RREQ from an upstream node, a node inserts a

reverse route to the upstream node in its reverse routing table if it does not have such a route. If it has such one route but the route's hop count is greater than the one in the RREQ and the route's sequence number is equal to that of the RREQ, the node updates the route entry about the upstream node in local reverse table. If the sequence number of the RREQ is larger than the reverse route's, the node updates the reverse route regardless of the hop count because it thinks that the reverse route is stale.

Similarly, the route update also occurs in forward routing table during the RREP relaying. If a relay node notices that its sequence number of a route to the destination of the RREP is less than the one in the RREP, it directly replaces with the route in the RREP. But if the two sequence numbers are equal, the node compares existing hop count with the one in the RREP to decide if to update the corresponding route.

Connectivity failure is another factor leading to route update. If a node no longer can track a neighbour by the HELLO message, it firstly finds out the routes including this neighbour in routing table and marks them as invalid. Then it broadcasts a RERR message about the neighbour to other ones. When another node receives this error message, it deletes all the routes containing that failure node from routing table. It then upwards unicasts the error message to the neighbours who forwarded RREQ messages to it before so that they update relevant routes.

2.4 Objectives of Wireless Mesh Network Security

As mentioned above, the routing protocols play an important role in solving some unique challenges of WMN. Unfortunately, the initial design of routing protocols for WMN assumes the network is the secure environment so the security has not been taken into account. In practice, if once the protocols are compromised by a malicious intruder, those unique challenges of WMN will be difficult to handle.

So the security issue in the WMN cannot be ignored. For the WLAN technology, basic security objectives can be characterised in terms of confidentiality, authenticity, integrity and availability. Because the WMN technology derives from WLAN, the security of WMN can also be measured at these perspectives.

1) Availability

Availability assures that information will be accessible for others whenever and wherever required.

2) Confidentiality

Confidentiality means that information being transferred through the network should be readable only to the intended recipient.

3) Authenticity

Authenticity warrants that received information originates from the intended sender, i.e., verification about the identity of the data.

4) Integrity

Integrity guarantees that received information is identical with what the intended sender actually transmitted without being modified by the third party in transit.

2.5 Classification of Attacks in Wireless Mesh Networks

As discussed earlier, the characteristics of WMN are contributed to the unique challenges, such as vulnerability to attacks, limited available channel capability, multi-hop routing and so on. Among these issues, the realm of security for WMNs needs to be paid more attention.

Although both wired and wireless networks face the security threats, wireless networks suffer from a number of unique threats which wired networks are generally not required to deal with. For example, an attacker is able to easily intrude into a WMN just by being in range of the network and intercept transmission data through the air. But this situation hardly occurs in wired networks because of the physical protection to the transmission media.

The attacks in wireless networks can be classified by a number of different criteria. Based on the domain/sources of attacks, the attacks in wireless networks are classified into insider attacks and outsider attacks[9]. The insider is someone who has been entrusted by the network, and also may have knowledge of the network

topology. On the contrary, the outsider, who attempts to acquire access to an authorized identification, does not pertain the domain of the network.

Alternatively, security attacks against the WMN are typically divided into passive and active attacks[10] in Figure 2-4. Then these two broad classes can be subdivided into four types of attacks on the basis of security services, i.e., availability, confidentiality, authenticity and integrity.

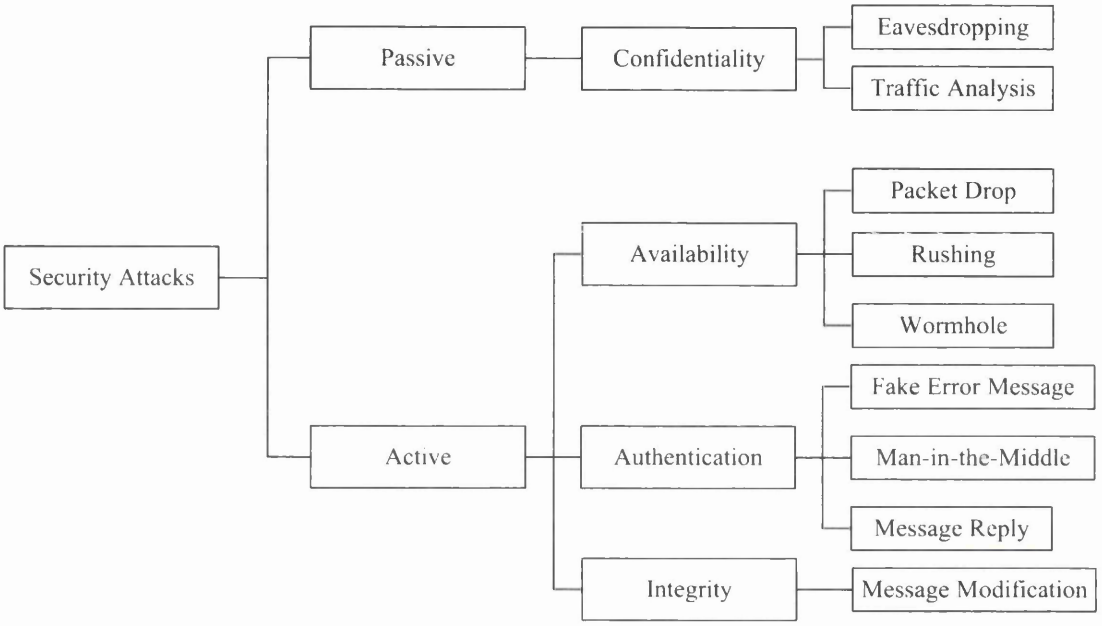

Figure 2-4 Classification of Typical Attacks in WMNs

2.5.1 Passive Attacks

A passive attack is the one in which the intruder intercepts but does not modify messages in any way. In practise, the passive attack usually takes the form of a combination with an active attack described in Section 2.5.2.

1) Attacks on Confidentiality

Attacks on confidentiality are harmless as long as they did not combine with those active attacks. In this kind of attacks, the attacker audits all the data that is exchanged between its neighbouring nodes, and then gets content of data packets, locates some nodes that are vital for the functioning of other nodes. Once these sorts of information are known, the attacker can successfully launch any of active attacks. Generally speaking, there are two types of threats against the confidentiality:

eavesdropping and traffic analysis.

- Eavesdropping

A eavesdropping attacker monitors wireless data transmissions between nodes for obtaining message content, such as passwords[10]. An example of this attack is an attacker listening to transmissions between the AP and the mesh client in the WMN.

- Traffic Analysis

In traffic analysis, attacker can deduce some information by monitoring the transmissions for patterns of communication. In general, the greater the number of messages observed, or even intercepted and stored, the more can be inferred from the traffic[11]. In particular, the adversary is able to locate a particular AP by analysing the flow size of traffic. Because the mesh clients nearer the mesh AP forward a significantly greater number of packets than those further away from the AP, in the same manner that a river grows wider as it collects more water from its tributaries[12].

2.5.2 Active Attacks

An active attack is the one in which the intruder may transmit messages, replay old messages, modify messages in transit, or discard selected messages. It aims to compromise the network security in terms of availability, authenticity and integrity.

1) Attacks on Availability

- Packet Drop

A packet drop attack is a type of attack where a router or an intermediate node should have forwarded packets, but discards them instead. In a WMN, if a node drops all of incoming packets including routing messages and data packets, it has minor effect on the network because it will not be selected as a relay to forward packets. It seems that this isolate node does not exist in the network.

However, if a malicious node takes participation in routing activities and claims having the shortest route to a destination but actually refuses to relay data packets, all traffic will be directed to the node and be discarded in accordance with its will.

Consequently, such an abnormal behaviour will lower the availability of services of the network.

What is worse, sometimes a malicious node accomplishes packet forwarding from time to time, i.e., only dropping packets for a particular network destination, forwarding a packet every *n* packets, or randomly selecting packets to forward. This is so called Grey-hole attack. Such type of attack is often not straightforward to be detected and distinguished because network traffic still passes through the malicious node and accidental link failure also appears to drop packets.

- Rushing Attack

To limit the overhead of transmission, routing protocols for ad-hoc networks generally allow each node to forward only one RREQ originated from any Route Discovery process. Moreover, each node only forwards the first arrived RREQ from each Route Discovery and discards the following ones. For this kind of property, in rushing attack the malicious node forwards RREQ faster than any other normal nodes in a network and hence ensures that it can be chosen as a relay in the route to a destination. Once such attack achieved, the malicious node is able to successfully launch the above packet drop attack.

- Wormhole

In a wormhole attack, after a wireless or wired link (called as wormhole link) is established, two malicious nodes (wormholes) can replay each other's messages in their own radio range. As a result, a route passing through the wormholes seems to be the shortest path between a source and a destination node.

An example is shown in Figure 2-5 where X and Y are the two wormhole nodes. X replays in its neighbourhood (area A) everything that Y heard in its own neighbourhood (area B), and vice versa[13]. Consequently, the node S in area A regards the node D in area B as its neighbours, and vice versa. So packet stream originating from or forwarded by node S flow towards destination D are guided to the way of wormhole link. In this situation, the wormhole node X can start packet drop attack leading to network disruption.

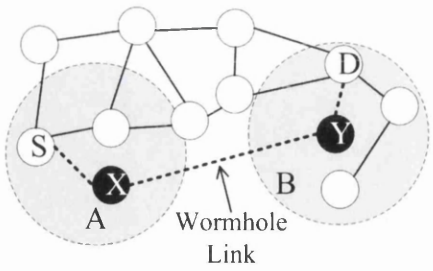

Figure 2-5 Wormhole Attack

2) Attacks on Authenticity

- Man-in-the-Middle

The Man-in-the-Middle attack is a kind of active eavesdropping attack resulting in that two communicating parties believe that they are exchanging information with each other but not so in fact. For instance, in Figure 2-6, at the beginning the two nodes A and B have established an original communication connection allowing them communicating directly. But later the malicious node M impersonates B to reply A's route request, and vice versa. At last, M makes the two victims think that they are still connected with each other directly. Consequently, the messages that A sends to B will be relayed by M, and vice versa. That is, since then M can read or modify the content of conversation between A or B.

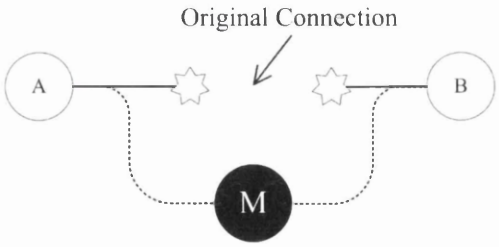

Figure 2-6 Man-in-the-Middle Attack

- Fake Error Message

An attacker may fabricate and propagate a false Route Error message over a network on behalf of a particular node, which causes other nodes to remove this node from their routing table. As a result, these nodes continuously initiate new route request messages, and thus unnecessarily consume resources: transmission capability and energy.

- Message Reply

In a message replay attack, an attacker eavesdrops and obtains a copy of an encrypted message sending to a particular recipient, and in a while the attacker retransmits the captured message on behalf of the original node of the message in attempt to be accepted by the recipient.

3) Attacks on Integrity

- Message Modification

All the nodes send, receive or forward data packets over ad-hoc networks, depending on their route entries. Message modification attacks aim to alter the routing message (e.g., route request or route reply) that are available to every node in the network.

Take AODV as an example. An attacker may change vital information, such as sequence number and hop count, contained in AODV routing messages to cause incorrectly entries update in other nodes' routing table. In this way, the attacker could be prone to be selected as a relay and attract data traffic over the network, which is convenient for the attacker to start the packet drop attack.

2.6 Security Countermeasures

2.6.1 Information Encryption

A technical method of enhancing confidentiality, authenticity and integrity protection is using the two schemes: public-key and digital signature both based on encryption algorithm.

1) Public-key

The public-key scheme is different from the traditional and intuitive cryptographic mechanism known as private-key where both the message sender and receiver share the same secret key. The shared secret key must be distributed to them in a safe manner. Once an attacker learned one key, he or she can crack all encrypted information afterwards.

Therefore, to address the drawbacks of private-key, the first public-key method was

developed at the British Government Communication Headquarters (GCHQ)[14]. In this method, users can communicate over a public channel without having to agree on a common shared key beforehand.

The public-key scheme requires a pair of keys: a public key and a private key. The first one is to encrypt a message. The other is to decrypt the cipher message. Figure 2-7 illustrates how they work together. First, Bob disseminates his own public key to everyone, e.g., Alice, in his communication network. Then, if Alice intends to transmit a message to Bob, she encrypts her message using the received public key of Bob. After Bob received Alice's cipher message, he uses its own private key to decrypt the cipher message for reading.

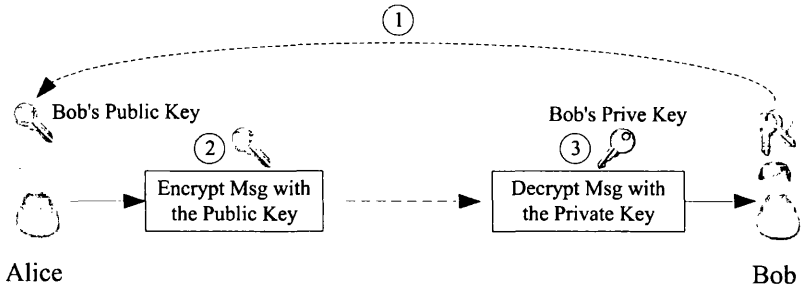

Figure 2-7 Public-key Scheme

The premise of this cryptography scheme is that it is easy to compute the pair of keys but, reversely, difficult for anyone to figure out the private key based on one's knowledge about the public key. Under this condition, messages from Alice are only readable for Bob, instead of others. Obviously, the confidentiality of messages is protected by the public-key scheme.

2) Digital Signature

In addition to the encryption, the public-key cipher algorithm can also be applied into the digital signature scheme to ensure the authenticity and integrity of a message. An example can be seen from Figure 2-8.

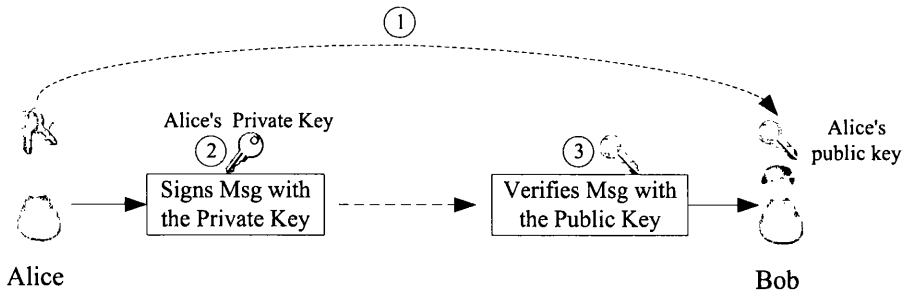

Figure 2-8 Digital Signature Scheme

Firstly, to guarantee that Bob will receive the genuine message from her, Alice propagates her public key to him in advance. After that, she signs a message with her own private key. The signature is a snapshot of the message content. At last, Bob receives this signed message and verifies the signature using the public key of Alice to check the authenticity and integrity of the message.

Notice that the cryptography is valid for overcoming the threats to the confidentiality, authenticity and integrity but not ideal for the availability.

In addition, the public-key cryptography uses a mathematical "one-way function" to generate the keys, which has been seemed as sufficiently secure for decades. However, with technological progress, some wireless network software (e.g. Cain & Abel) can crack a number of cipher algorithms (e.g., RSA) based on one-way function to recover encrypted information. By such techniques, an attacker can be trusted by other nodes, gain the wanted information, and launch attacks in a WMN.

So, to deal with this adverse situation, the intrusion detection should be introduced into WMNs for determining whether malicious users or devices have already intruded and compromised some nodes.

2.6.2 Intrusion Detection

As another line of defence, Intrusion Detection (ID), described in next chapter, is a type of security technique for wired and wireless networks. The Intrusion Detection System (IDS) gathers and analyzes information from various areas within a network to identify possible security threats.

2.7 Summary

This chapter first presents the unique challenges caused by the characteristics of wireless mesh networks. Then it concentrates on the security issues faced by WMNs. In particular, it categorises the attacks on routing protocols in accordance with security objectives of wireless networks. At last, it gives out corresponding countermeasures to the threats on confidentiality, authenticity and integrity, and introduces the intrusion detection overcoming the adverse situation that an attacker has intruded and been trusted by a network.

3 Intrusion Detection

Since the recent years have seen a dramatic increase in attacks against Wireless Mesh Networks, intrusion detection is proposed for detecting the actions that attempt compromising the security of a resource[15]. Intrusion detection can be viewed as a passive defence[16], like the burglar alarm in a building which responds after an abnormal event happened. A system that performs intrusion detection is called the Intrusion Detection System (IDS).

Even though IDS in WMNs has a number of similarities to its counterpart in wired networks in terms of system structures and analysis approaches to attacks, intrusion detection techniques applied in wired networks cannot be directly transplanted to WMNs in view of their decentralized nature, dynamic network topology, and easy access to the radio medium for attackers[16].

As well as introducing the general architectures and analysis approaches of the IDS, this chapter investigates existing intrusion detection and reputation schemes for WMNs in order to show the feasibility of a reputation based intrusion detection scheme for packet drop attack.

3.1 Intrusion Detection Architecture

Typically, there are three types of IDS. The first type is host-based system monitoring a host's system files and packets passing through the host to detect the abnormal events. The second one is network-based system which is placed at strategic points within a network to provide maximum inspection of all network traffic[17]. The last one is a combination of the two types of IDS.

3.1.1 Host-based Detection

Host-based Intrusion Detection System (HIDS) is a typical example of independent detection. It records internal events of a system into log files and periodically scans these files for abnormal activity. If a suspicious event is detected, a notification will

be emitted. Another host-based approach monitors all incoming and outgoing packets on the host's network interfaces.

It is admitted that the HIDS can tell if a potentially malicious behaviour actually happened at the current host due to its overall scope of log files about the local system. In addition, the host is the best position to counter any attacks in a network. So it is appropriate for protecting an individual host from being intruded.

However, the HIDS has infrequent interaction with outside or does not provide any data for others in the whole network. Also, compared with its counterpart Network-based Intrusion Detection, it consumes more processing resource of the local host.

3.1.2 Network-based Detection

Network-based Intrusion Detection System (NIDS) monitors and collects network traffic to analyses the purpose of the traffic. So far, the basic automatically analysis methods towards attacks used in NIDS are generally classified as knowledge-based (misuse) detection and behaviour-based (anomaly) detection.

1) Knowledge-based Detection

Knowledge-based detection is also known as misuse or signature detection because it relies on a single or composite signature representing characteristics of an attack. A signature could be created based on the content or header of a packet. Signatures can be described as a boolean relation called rule[18]. If a matching between signatures and observed traffic is found based on past experience and rules, an alert of intrusion behaviour is to be emitted.

An advantage of signature detection is its accuracy. If a signature matches, that signature identifies a specific attack. Knowledge of the specific type of attack means that an appropriate response can be determined immediately[16].

However, knowledge-based detection has several limitations. First, a slight change in the attack scenario is possibly enough to alter the attack signature and thus fool a signature filter[18]. Second, such detection can only recognise those specific attacks

whose corresponding signatures have formed. That requires continuous work of updating signatures of the new attacks that have been evolving. Consequently, knowledge-based detection could not detect a number of new attacks even though it has a low rate of the false positive error in which a non-match is declared to be a match.

2) Behaviour-based Detection

Behavior-based detection, also named anomaly detection, is totally different from knowledge-based detection. It discovers intrusion behavior by the patterns of normal network traffic. If the history of an activity does not resemble any normal pattern, the activity can be assumed to be abnormal. (On the contrary, knowledge-based detection tries to characterize attacks, and everything else is assumed to be normal[16].) The patterns are constructed by statistical analysis of the observation about past normal behavior. Once the patterns have been set up, intrusion activities can be recognised from the normal ones whose statistics patterns deviate from those of malicious behavior.

A major advantage of behavior-based detection is the potential to detect new attacks without prior experience. That is, a signature for a new attack is not required; a new attack will be recognised if it significantly deviates from the normal behavior[16].

In the other hand, the accuracy of this detection depends on the deviation between the patterns of the normal and abnormal activity. If the deviation is not enough significant, the behavior-based detection will be vulnerable to high rate of the false negative error (an actual match is not detected). Besides, an event assumed anomaly is not necessarily malicious.

3.1.3 Hybrid Detection

In practice, to overcome the limitations of purely Host or Network based IDS, and to combine the strength of them, researchers have proposed Hybrid Intrusion Detection System (HIDS), which allows user to mix the signature based and the anomaly based strategies[19].

3.2 Intrusion Detection for WMNs

Due to the open and shared nature of the wireless medium mentioned in Section 2.2, intrusion detection techniques for traditional wired networks cannot be copied over to WMNs. Additionally, the autonomous property of mesh nodes requires intrusion detection schemes should be decentralized. Therefore, new distributed intrusion detection schemes must be designed for the WMN[16].

3.2.1 Distributed Intrusion Detection

The WMN is distributed by nature and requires nodes to be cooperative. A survey of intrusion detection schemes proposed for WMNs is given out in this section and most of them are tightly related to routing protocols. These schemes share some common concepts but differ in the details. Additionally, the shortcomings of each scheme are described as well.

1) Cooperative Anomaly Detection

Zhang and Lee[20] developed a distributed and cooperative anomaly detection scheme for the MANET, and evaluated such scheme with the DSR[21] and AODV routing protocols. From the perspective of functions, the scheme consists of three functions: monitoring, analysis and response. The basic of this scheme is that each node is responsible for detecting signs of intrusion independently. If necessary, neighbouring nodes can collaboratively investigate an activity in a broader range.

Specifically, the function of detection is in the charge of an individual agent at each node. Each agent audits local events such as systems activities, and communication activities within its radio range. If it has strong confidence in a decision about an intrusion activity based on local traces, it will respond to the suspicious activity individually. But if it has an inconclusive evidence of an anomaly, a broader detection will be triggered: IDS agents at its neighbouring nodes will together involve in performing global intrusion detection and response. These individual IDS agents collectively constitute the IDS defending attacks in MANET.

The anomaly detection is utilized as intrusion analysis approach in the scheme. Based on information-theoretic measures, entropy and conditional entropy are used

to describe the characteristics of normal information flows. And two classification algorithms (Decision-tree and Support Vector Machine, SVM) are responsible for building anomaly detection models for the attacks on routing protocols. It is also discussed in [20] that such analysis mechanism can be implemented in other layers of the wireless networks.

Intrusion response depends on the type of intrusion, the type of routing protocols and applications, and the confidence in the evidence[20]. One common intrusion response in this proposed scheme is re-computing a route to avoid compromised nodes. The other response action is to re-authenticate two nodes connecting to a suspicious link. The former one is more often used because the performance of the latter is subject to authentication credentials. For an insider attacker, the re-authenticating measure is not effective because it has been trusted by the network.

2) Watchdog and Pathrater

Marti et al.[22] described and applied two techniques, watchdog and pathrater, into the DSR routing protocol for mitigating the effect of the node that agrees to forward packets but fail to do so. The experimental results show that the two techniques can minimize the effect of this type of misbehaviour while increasing the throughput of the network.

The watchdog is a process running at each node to monitor the behaviour of the next hop. Figure 3-1 illustrates how the watchdog works. Suppose that S, having a route to D via intermediate nodes A, B and C, attempts to send a packet to D. Before A forwards the packet for S, it first stores the packet in a buffer. Once the packet is forwarded, the watchdog at node A monitors the next hop B to ensure whether B forwards the packet to C. If a forwarded packet from B is captured by A (shown by dotted lines) and the packet is also identical to what has been stored in the buffer of A, it then can be deemed that B has successfully forwarded the packet to C (shown by solid lines) on behalf of A. And then the packet buffered in A is removed. Otherwise, if no captured packet matches the buffered packet within a specific time, the watchdog will increase the failure counter of node B.

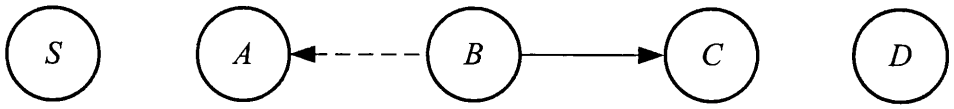

Figure 3-1 Watchdog Operation

The pathrater computes a path metric for every path to avoid malicious node. Specifically, each node rates other nodes it knows between 0 and 1. The path metric is calculated by the averaging rating of every node along a path and the connection reliability obtained from previous experience. If there are more than one path to the same destination, the path with the highest metric will be selected while another path including misbehaving nodes is avoided.

Although the two techniques are capable of mitigating the effect of the misbehaviour, this scheme also brings into the following weaknesses. First, it approximately doubles the transmission overhead within scenarios of moderate mobility and extreme mobility, respectively. Second, the rating of the watchdog is obtained by monitoring the traffic through its host node without referring to any external opinion, so the result is thus prone to be one-sided. Thirdly, calculation about the path metric is based on that the pathrater is clear about the route which a packet passes through. This means that the calculation is completely performed by the sender of the packet. Lastly, a misbehaved node could be detected until its dropping packets surpass a given threshold.

3) Brief Comparison

The basic idea of the above intrusion detection schemes is monitoring the packet traffic within radio range. They audit the network traffic by the IDS agent at each node and the watchdog, respectively. Zhang and Lee[20] did not illustrate the data collection process in detail, they just defaulted that network data could be accessed through a certain way. But Marti et al.[22] presented that the watchdog could complete such collection work.

With respect to functions, the IDS agent is more complex than the watchdog. The agent can make a decision for responding to a suspicious behaviour by itself. But the watchdog only concentrates on packets monitor and analysis, not considering related response.

Seen from the perspective of analysis approach to intrusion, the IDS agent utilizes behaviour-based detection method to recognise normal and anomalous activity. It develops a pattern of normal activity by a SVM classifier which needs training data beforehand. On the contrary, the watchdog is only designed for detecting behaviour of dropping packets. To achieve this, the watchdog analyses the captured packet instead of requiring pre-processed data to build a pattern.

From the view of cooperation, a cooperative detection is requested when the IDS agent has not enough confidence in its detection result. A requesting node shares its data about the suspicious event with its neighbouring nodes, and all of them follow a consensus algorithm to determine whether to generate an alert in a global scope. In contrast, Marti et al.[22] did not consider such a global decision because they only focused on detecting the packets dropping behaviour and assumed that the misbehaviour can be judged by a node itself independently.

Note that, decision-making relying on the cooperative participation of nodes can be easily misguided by false information[23]. If without any countermeasure, malicious nodes would spread false information over a network to guide the IDS making a wrong decision. Therefore, the IDSs, presented in the next subsection, include reputation mechanism to minimize the effects from malicious nodes.

3.2.2 Reputation-based Intrusion Detection

As mentioned in Section 2.5, attacks in wireless networks can also be divided into the insider attack and the outsider attack. While most outsider attacks such as spoofing network protocols can be avoided by authentication and encryption schemes, insider attacks (e.g., packet drop attack) are relative harder to deal with[24]. Trust and security are two tightly interdependent concepts[25]. It is natural to expect avoiding inside attacks by combining a reputation mechanism into the intrusion detection.

Besides, limited wireless transmission range results in partial vision about the traffic in wireless networks. To synthesise different local opinions of various nodes into an accurate result, every node should be allocated to a specific weight standing for its

reliability. Virtually, this weight could be a reputation value that is offered by the reputation mechanism.

In addition, an external node can join a WMN arbitrarily, as long as it enters the radio range of the network. Obviously, malicious nodes are able to easily merge into the WMN and to negatively influence the behaviour or data of the normal ones. Moreover, a wireless node with limited range is likely to suffer from slow and dropped connections. So, unlike the wired link, wireless link cannot continuously ensure acceptable stability and reliability. As a result, one's opinion may be dropped or distortedly transferred to others.

In view of the local vision and the vulnerability of wireless entities, nodes in the WMN cannot be entirely trusted. Therefore, reputation mechanism should be considered as a component for intrusion detection, which provides a means to identify misbehaved nodes. This section reviews typical reputation schemes to show the state of the art.

1) CONFIDANT

The CONFIDANT[26] scheme aims at detecting and isolating misbehaving nodes. The implementation of this scheme assumes that the network layer routing protocol is based on DSR. In this scheme, each node can also observe the behaviour of all its neighbouring nodes. The authors believe that a misbehaved node should be punished in a certain way rather than, like the Watchdog and Pathrater, be prohibited from forwarding packets to others. Also, when discovering a misbehaved neighbour, a node informs all other nodes not to forward packets through this neighbour as well.

In particular, the CONFIDANT system consists of Monitor, Reputation System, Trust Manager and Path Manager. The monitor enables each node to audit the behaviour of its neighbouring nodes. If an activity of its neighbour is seemed as suspicious, the relevant information will be passed on to the reputation system of the monitoring node. If this activity deviates from normal ones significantly, the monitoring node checks whether such the activity has occurred more often than the predefined threshold that is used to differentiate deliberate malicious behaviour from occasional coincidences such as collisions. If the threshold is exceeded, the

reputation system gives a negative reputation rating to the monitored neighbour. If the reputation of the neighbour decreases to a rather low level, the neighbour's information is taken over by the path manager that removes all the routes containing the suspicious neighbour from a path cache. In this way, the path manager can avoid to select a route including the misbehaved node. Moreover, an alarm message is sent by the trust manager of the monitoring node to warn others against the suspicious node. The precondition of the whole process is that the monitoring node is trusted or several negative reports about the neighbour are received by the trust manager.

Since this scheme allows network nodes to send alarm messages to each other, it is therefore a good opportunity for the attackers to send false alarm messages about normal neighbours to a node in order to misguide its decisions[27].

2) CORE

CORE[28] enforces cooperation among nodes in a MANET to prevent selfish behaviour. Each node keeps the track of the other nodes' collaboration and represents a degree of the collaboration with the reputation that is based on the monitor in CONFIDANT. In CORE, each node receives positive reports from others and calculates the reputation based on their rate of collaboration.

The main difference between CORE and CONFIDANT is that CORE does not pass negative ratings about other nodes through the network but CONFIDANT allows negative reports about others. Therefore, attacks broadcasting negative ratings for legitimate nodes, which CONFIDANT suffers from, are prevented.

CORE also differs from the Watchdog-Pathrater scheme. CORE was proposed as a generic mechanism that can be integrated with several network and application layer functions, even though it is only applied to monitor packet forwarding function of DSR using the watchdog[28]. But the Watchdog-Pathrater scheme is specifically designed for routing protocols.

It should also be pointed out that both CONFIDANT and CORE assign the same weight to each neighbour's rating. A more reasonable scenario should be that the rating from a normal node is given more weight than that from a malicious one and

thereby eliminating the interference from the misbehaved node.

3) SORI

Similarly, the SORI[29] scheme also uses the reputation information as a measure to counter the selfish behaviour of a node.

Specifically, each node updates its local evaluation record for all its neighboring nodes periodically. If the reputation of a node in the context of forwarding packets changes significantly, the monitoring node informs its neighboring nodes about the change. Moreover, if a node receives reputation reports from multiple nodes, it aggregates these observation results and weights them by reputation of the sending nodes. Furthermore, since a node broadcasts an alarm about a misbehaved node to its neighbors when the reputation value of the abnormal node falls below a threshold, the overhead is significantly less than the CONFIDANT scheme.

4) REFACING

REFACING[5] focuses on achieving a distributed anomaly detection model by utilizing the Dempster-Shafer[30] evidence (D-S) theory to combine detected data and to figure out a comprehensive reputation for a given node. Based on the reputation, the overall detection accuracy can be improved. In Figure 3-2, the REFACING system is composed of four separate functional parts: Detection, Normalization, Data Fusion, and Reputation Update.

The detection nodes are used to detect potential malicious behaviour in a network. Some specific events (e.g., a timeout for a buffered packet) will proceed to a verdict according to the detection node's inner judgement. Then, the verdicts from different nodes are normalized in order to have a uniform view of the various responses. After that, the normalized data are processed by the combination rule of the D-S theory[31] to obtain a comprehensive judgement about the network. The difference, between the final overall verdict and the verdict of the detection node, is used to measure the level of agreement or disagreement between the network and the local node. The difference is also applied to update the reputation weight of the detection node. A node's updated reputation has an influence on its verdict in the next detection loop.

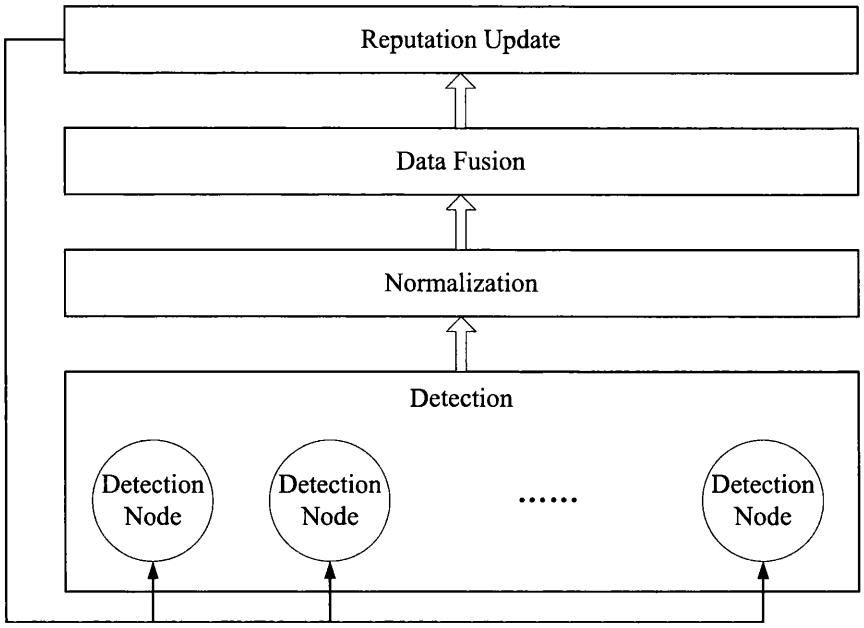

Figure 3-2 Function Framework of REFACING

A breakthrough of this scheme, compared with the previously mentioned, is that it applies the information fusion algorithm to minimize the effects of false information. But it also suffers from the D-S's inherent weakness presented in Section 4.1.3.

Besides, the authors[5] just concentrated on illustrating the applicability of this scheme rather than on the capability to detect specific attacks. Furthermore, the papers[32, 33] simulated and implemented the Reputation Update function in AODV to compute a reputation metric for mitigating the effects of packet drop attack, respectively.

One weakness of the above reputation mechanisms is that a node's reputation is not unique over a given network and depends on a specific route. Although one node can make a conclusion based on the opinions from others, different nodes generally have various reputation evaluations toward the same node. As a result, a misbehaved node may enjoy a high reputation in some node's eyes.

5) SOPE

In order to evaluate a node's reputation for intrusion detection independent of any specific route or path, SOPE[34] applied the EigenTrust algorithm in DSR to calculate a consensus reputation for every node from the perspective of a whole network.

In particular, this scheme assigns every node a new property called 'centrality'. The nodes with both high centrality and high reputation are more likely to become the sources for the reputation iteration. In fact, the sources play the same role as the pre-trusted nodes in the original EigenTrust, which guarantee that every node has a certain probability to converge and an ability to resist against the interference from malicious nodes. In addition, SOPE defines a 'degree centrality' to allocate higher weights to the highly central nodes (sources), which leads to fast reputation convergence. Actually, what the degree centrality increases is just the value of α in equation (4-15). Because both the introduce of pre-trusted nodes and the increase of α enlarge the second principle eigenvalue (λ_2) of a network[35] and the convergence rate of EigenTrust depends on the ratio of λ_2 and the principle eigenvalue λ_1 of the network, the convergence rate in SOPE becomes fast.

However, if the high centrality node is selected improperly, SOPE cannot ensure the algorithm convergent. Therefore, Chapter 4 improves the EigenTrust algorithm to guarantee its convergence when no pre-trusted nodes are specified, and Chapter 5 implements the improved algorithm into AODV.

3.3 Intrusion Tolerance

Regardless of HIDS or NIDS, they should be robust enough to run continually in the background without human intervention[19], even if a malicious insider resides. For this reason, intrusion tolerance is proposed by researchers.

Different from the intrusion detection, the design concept of intrusion tolerance assumes that it is impossible to develop a general protection mechanism against attacks and some of them will achieve, because not all attacks are well characterized and there are always unknown attacks in practice. So, in fact, intrusion tolerance is a fault-tolerant approach to protect networks against malicious behaviour.

For a reputation-based intrusion detection, the importance of an inside attacker's opinion is determined by its reputation and the other entities' evaluation. Thus, if only a certain amount of normal entities are compromised, with help of the reputation mechanism, the impact of the malicious intruder will have minor impact

on the security of the network. This reputation-based approach is thus intrusion-tolerant.

3.4 Detection Solution for Packet Drop Attack

Routing protocols are highly vulnerable to be attacked. Countermeasures for limiting the effect of attacks on routing protocol are thus becoming a mandatory requirement for the WMN. Especially, the packet drop attack seriously threatens the security (resource availability) and compromises the performance of the network.

Additionally, the continuous growth of WMNs calls for particular solutions to their security. Recent research efforts have been put on how to achieve a distributed, cooperative and effective intrusion detection scheme for the WMN. As a result, information fusion algorithms are applied to combine data from multiple intrusion detection points distributed over the network, as shown in [5, 32].

Therefore, referring to the function framework shown in Figure 3-3, the following chapters focus on a distributed and cooperative intrusion detection scheme that utilizes EigenTrust[4] to synthesise a comprehensive trustworthiness for every node in WMNs. The trustworthiness is mapped to a reputation metric of AODV for detecting the nodes compromised by the packet drop attack.

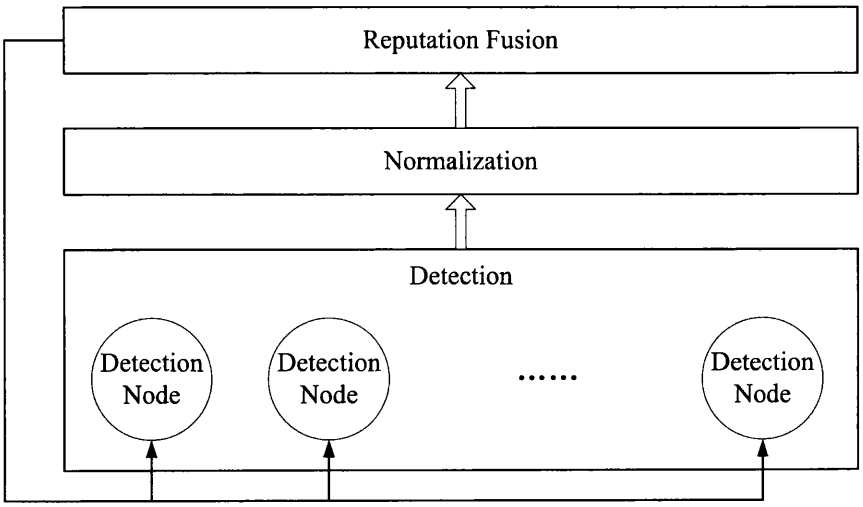

Figure 3-3 Simplified Function Framework of Cooperative IDS

3.5 Summary

As a conventional defence scheme in wired networks, intrusion detection cannot be simply borrowed from wired networks to wireless networks because of those discussed challenges inherent in wireless media. But the traditional detection system architectures and approaches presented in the first section of this chapter are still valid for the wireless environment.

Besides, recently published research on the distributed, cooperative and reputation based intrusion detection schemes, which enhances the attack tolerance of wireless networks, is presented in this chapter too.

Finally, the common shortcoming of existing reputation mechanisms is pointed out and the functional framework of intrusion detection is simplified in Figure 3-3, which will be associated with the EigenTrust algorithm described in Chapter 4 to calculate the reputation against the attacks dropping packets.

4 Reputation Fusion for Intrusion Detection

One challenge in intrusion detection is to correctly judge whether the behaviour of an entity in a network is normal or not based on the detected data. Ideally, the IDS can eliminate two types of errors: false positives (a non-match is declared to be a match) and false negatives (an actual match is not detected). Unfortunately, it is unlikely to avoid both errors. The IDS can only attempt to minimize both types of error through some methods, e.g., relying on decisions made by reliable nodes and data fusion.

Indeed, information collected from a single source is usually limited and sometimes provides for low accuracy[5] or inconsistent results. For instance, methods of reputation calculation in many reputation mechanisms proposed in Section 3.2.2 cannot compute a consensus reputation in the scope of a whole network such that different nodes generally hold various evaluations about a given node. On the other hand, detection from multiple sources benefits from a wider field of vision. But it sometimes suffers from the dependence and the conflicts among the detected data[36], like the applications[37, 38] of Dempster-Shafer evidence (D-S) theory that is a solution of data fusion. In view of the above issues, a distributed reputation algorithm named EigenTrust[4], which can compute a comprehensive and consensus reputation for a node, is improved in terms of its initial value and termination condition in this chapter to implement it into AODV.

4.1 Dempster-Shafer Evidence (D-S) Theory

Dempster-Shafer evidence (D-S) theory is a kind of reasoning algorithm based on the evidence theory. It was proposed by Dempster at first and then developed by Shafer, so it is called D-S evidence theory[31]. It is used to combine evidence (opinions) coming from multiple sources under a premise that all evidence is independent.

4.1.1 Introduction of D-S Theory

Suppose that Ω is a sample space, consisting of a number of mutually exclusive subsets. In this theory, a mass function $M(\bullet)$, also named basic probability

assignment (*bpa*), expresses the proportion of all relevant and available evidence that supports a subset. $M(\bullet)$ meets the following description:

$$\begin{cases} M(\bullet) \in [0,1] \\ M(\phi) = 0 \quad \text{, where } \phi \text{ is an empty subset and } A \text{ is a subset of } \Omega. \\ \sum_{A \subseteq \Omega} M(A) = 1 \end{cases}$$

Besides, the D-S theory believes that the probability of an event A, $P(A)$, falls into an interval bounded by two measures called *Belief* and *Plausibility*, namely:

$$0 \le Bel(A) \le P(A) \le Pl(A) \le 1 \tag{4-1}$$

where the lower bound $Bel(A)$ is defined as the sum of basic probability assignments of all subsets of A, standing for the whole belief degree of the event A. The function $Bel(\bullet)$ is defined as follows:

$$\begin{cases} Bel(\bullet) \in [0,1] \\ Bel(\phi) = 0 \\ Bel(A) = \sum_{S \subseteq A} M(S) \end{cases} \tag{4-2}$$

where S is a subset of A.

The upper bound, $Pl(A)$, represents the belief degree of evidence not refusing A. It is the sum of basic probability assignments of all of Ω's subsets that intersect with A. The function $Pl(\bullet)$ is denoted as:

$$Pl(A) = 1 - Bel(\overline{A}) = 1 - \sum_{B \cap A = \phi} M(B) \tag{4-3}$$

where $\overline{A} = \Omega - A$ and $A, B \subseteq \Omega$.

4.1.2 Combination Rule

The Dempster's evidence combining rule provides a principle to combine two pieces of independent evidence. Specifically, the combination $M_{12}(\bullet)$ is calculated in the following manner:

$$M_{12}(A)=\begin{cases}\dfrac{\sum_{B\cap C=A} M_1(B)M_2(C)}{1-K}, & A\neq\phi \\ 0 \qquad\qquad ,otherwise\end{cases} \tag{4-4}$$

$$K=\sum_{B\cap C=\phi} M_1(B)M_2(C) \tag{4-5}$$

where $A,B,C\subseteq\Omega$. If the degree of conflict between B and C leading to K=1, the combination rule is invalid.

Let $Bel(A)=M_{12}(A)$ and according to (4-2) and (4-3) the confidence interval about the probability of event A can be decided. The particular computation process of this theory can be seen from the example explained in the next subsection.

4.1.3 Limitations

D-S combining rule of evidence can make correct decision if all evidence is independent and not highly conflicting, but when evidences are correlative or $K=1$ in (4-4), the combining rule may fail.

Let us consider a simple example of fault diagnosis[39] for illustrating the influence of evidence correlation on the final combining result. Suppose that there are three persons: an expert, a worker and a student, they diagnose the same system fault that is probably caused by event X, Y, or Z that is for a normal situation.

Their diagnosis results are as follows:

Expert (E): $M_E(X)=0.7$, $M_E(\Omega)=0.3$;

Considering the authority of the expert, the others make the same judgement, namely:

Worker (W): $M_W(X)=0.7$, $M_W(\Omega)=0.3$;

Student (S): $M_S(X)=0.7$, $M_S(\Omega)=0.3$;

Conversely, a reliable result given out by a testing instrument is $M_I(Y)=0.7$,

$M_I(\Omega) = 0.3$.

According to the combining rule of D-S theory, the diagnosis of expert and worker are firstly combined as shown in Table 4-1.

Table 4-1 First Fusion

	$M_E(X) = 0.7$	$M_E(\Omega) = 0.3$
$M_W(X) = 0.7$	0.49 (X)	0.21 (X)
$M_W(\Omega) = 0.3$	0.21 (X)	0.09 (Ω)

Thus, the fusion result of first step is

$M_1(X) = 0.49 + 0.21 + 0.21 = 0.91$
$M_1(\Omega) = 0.09$

After that, Table 4-2 combines the above results with the diagnosis of the student.

Table 4-2 Second Fusion

	$M_1(X) = 0.91$	$M_1(\Omega) = 0.09$
$M_S(X) = 0.7$	0.637 (X)	0.063 (X)
$M_S(\Omega) = 0.3$	0.273 (X)	0.027 (Ω)

Similarly, the fusion result of this step is $M_2(X) = 0.973$ and $M_2(\Omega) = 0.027$.

Again, the third fusion process with the instrument result is shown by Table 4-3.

Table 4-3 Third Fusion

	$M_2(X)=0.973$	$M_2(\Omega)=0.027$
$M_1(Y)=0.7$	0.6811 (ϕ)	0.0189 (Y)
$M_1(\Omega)=0.3$	0.2919 (X)	0.0081 (Ω)

Consequently, the final fusion result is as follows:

$M(X)=0.2919$
$M(Y)=0.0189$
$M(\Omega)=0.0081$
$M(\phi)=0$

It is impossible that an event stands by X and Y at the same time so $M(\phi)=0$ even though its calculation result is not so.

According to the D-S theory, the disagreement coefficient of evidence between people and the instrument is computed as $K=\sum_{X\cap Y=\phi} M_1(Y)M_2(X)=0.6811$. Then above results are normalized by coefficient $(1-K)$, thus:

$M(X)=0.2919/(1-0.6811)=0.9153$
$M(Y)=0.0189/0.3189=0.0593$
$M(\Omega)=0.0081/0.3189=0.0254$

Finally, based on the equation (4-2) and (4-3) the confidence interval of X is [0.9153, 0.9407], and the confidence interval of Y is [0.0593, 0.0847]. As a result, the system fault is seemed to be caused by the event X with a rather high confidence between [0.9153, 0.9407], but the correct diagnosis from the instrument is not considered, even though the instrument is generally more credible than the expert.

From the above discussion, the fusion result in the D-S theory is possibly unreliable when all of raters are believed equally and the correlation between their evaluations is ignored. Another example[40] shows that highly conflicting evidence also misguide

the theory.

Generally, indirect detection data coming from remote or unreliable nodes should be considered less trustworthy than that observed by adjacent neighbourhood. Therefore, it is necessary to assign higher trustworthiness to relative near observers having good performance records in a network. In addition, the D-S theory only combines two evidences every time. For the combination of multi-evidence, it infers a result by the combination of any two of them. In view of these, the EigenTrust algorithm is introduced to calculate a reputation for every node in a cooperative manner.

4.2 Centralized EigenTrust Algorithm

It is wildly accepted that a reputation mechanism can be applied to represent the reliability of nodes in networks. Reputation values in EigenTrust are classified into two types: local reputation and global reputation. Every local reputation at one node is an evaluation towards its neighbouring node. One's global reputation represents the node's trustworthiness over a whole network.

4.2.1 Local Reputation

1) Credit Rating

In the EigenTrust algorithm, a local reputation l_{ij} is defined as the sum of credit ratings of individual transactions which a node i has done with a neighbour j in i's observation. In other words, the node i observes and rates its neighbouring nodes and stores their credits (i.e., local reputation) that reflect i's opinion towards them.

This requires that i is able to count the number of satisfactory data transactions to j, denoting as sat_{ij}, and the number of unsatisfactory events, $unsat_{ij}$. Then, l_{ij} is computed as:

$$l_{ij} = sat_{ij} - unsat_{ij} \tag{4-6}$$

Obviously, if there were no connection linking the two nodes i and j, the local reputation would be zero, namely $l_{ij} = 0$ and $l_{ji} = 0$.

2) Normalization

In order to ensure the convergence of the algorithm, normalized local reputation $\bar{l}_{ij}$, defined by formula (4-7), is used in computing node j's global reputation rather than l_{ij} obtained by equation (4-6).

$$\bar{l}_{ij} = \begin{cases} \dfrac{l_{ij}}{\sum_{\substack{j=1,\\ j\neq i}}^{n} l_{ij}} , & if \sum_{\substack{j=1,\\ j\neq i}}^{n} l_{ij} \neq 0 \\ 1/n , & otherwise \end{cases} \tag{4-7}$$

where n is the number of all nodes in a network. This warrants that all local reputation is in the same interval [0, 1].

4.2.2 Global Reputation

In the social network, if a job seeker and an employer did not know each other before but the employer are familiar with the seeker's referrers, the employer would be likely to evaluate the seeker according to the opinions of the referrers and to weight their opinions based on their own reputation.

Similarly, in an n-node network, a natural way for the node i estimating another one k is that i asks its neighbours' opinions about the node k. Their opinions would also be weighted by the evaluation values from the perspective of i. This can be described by:

$$g_{ik} = \sum_{\substack{j=1\\ j\neq i,k}}^{n} \bar{l}_{jk} \cdot \bar{l}_{ij} \tag{4-8}$$

where g_{ik} represents a global reputation of a stranger k in the view of node i. $\bar{l}_{jk}$ is the local reputation about node k in j's eyes. Clearly, the node j is a broker between i and k. Extending the equation (4-8) to the entire network, we have a matrix notation:

$$\begin{bmatrix} g_{i1} \\ g_{i2} \\ \vdots \\ g_{ik} \\ \vdots \\ g_{in} \end{bmatrix}_{n\times 1} = \left[\bar{l}_{ij}\right]^T_{n\times n} \cdot \begin{bmatrix} \bar{l}_{i1} \\ \bar{l}_{i2} \\ \vdots \\ \bar{l}_{ik} \\ \vdots \\ \bar{l}_{in} \end{bmatrix}_{n\times 1} \tag{4-9}$$

where $\bar{l}_{ij}$ is set to 0 if the two nodes do not directly connect with each other. The matrix $\left[\bar{l}_{ij}\right]^T$ contains every node's evaluations about strangers, friends, and itself. One's subjective evaluation about itself should be ignored, so $\bar{l}_{ii}$ does not play any role in EigenTrust even though they are set to a number, e.g., 1. If the global reputation vector $\left[g_{i1} \quad g_{i2} \quad \cdots \quad g_{in}\right]^T$ and the normalized vector $\left[\bar{l}_{i1} \quad \bar{l}_{i2} \quad \cdots \quad \bar{l}_{in}\right]^T$ are denoted as $\vec{g_i}$ and $\vec{l_i}$, respectively, as well as the matrix $\left[\bar{l}_{ij}\right]^T$ is represented by L, then the global reputation vector of node i is presented by:

$$\vec{g_i} = L \cdot \vec{l_i} \tag{4-10}$$

As discussed in [4], equation (4-10) can only enable the node i to acquire an opinion about its neighbours' neighbour, rather than more distant nodes. To apply this equation into the entire network, the node i needs to ask its neighbour's neighbour for getting a wider vision, that is $\vec{g_i}^{(2)} = L \cdot (L \cdot \vec{l_i})$. If continuing to do in this way, the node i will have a complete view of the network after N iterations. The iterative expression is as follows.

$$\vec{g_i}^{(N)} = L \cdot \vec{g_i}^{(N-1)} = (L)^N \cdot \vec{l_i} \tag{4-11}$$

4.2.3 Constraints

In fact, the above sections describe the EigenTrust algorithm for centralized networks. Every node can figure out the global reputation vector $\vec{g}$ but a precondition is that the matrix $L_{n\times n}$ must be known. Otherwise, the calculation process cannot go

through. From another perspective, the precondition indicates that each node has known the topology of the entire network before the calculation is started. It is relatively straightforward to apply the centralized EigenTrust algorithm to proactive (table-driven) routing protocols because each node is responsible for maintaining a routing table including the routes to all destinations in a network. However, the situation is totally different if the algorithm is integrated with a reactive protocol, e.g. AODV, because only the information about next-hop and destination nodes is known to each node. Therefore, Section 4.3 shows a distributed EigenTrust algorithm, and modifications to the algorithm for applying it into AODV are illustrated in Section 4.4.

Additionally, the basic idea of the algorithm is to multiply the matrix $L_{n\times n}$ repeatedly by a given starting vector so that the vector converges to the direction of the matrix's eigenvector associating with the largest eigenvalue in absolute value[41]. To achieve this, an underlying assumption is that the square matrix $L_{n\times n}$ is irreducible and aperiodic. So the authors in [4] proposed equation (4-12) to transform $L_{n\times n}$ to an irreducible and aperiodic matrix for guaranteeing the convergence of vector $\vec{g}$.

$$\vec{g}^{(k+1)} = (1-\alpha)\cdot L\cdot \vec{g}^{(k)} + \alpha\cdot \vec{p} \tag{4-12}$$

where α is a decimal value between 0 and 1. The vector $\vec{p}$ represents a uniform probability distribution over P pre-trusted nodes, $p_i = 1/P$ if $i \in P$ (otherwise $p_i = 0$). Such nodes should be carefully selected. The following simulation results prove that the improved algorithm is convergent.

4.2.4 Algorithm Description and Simulation

1) Simple EigenTrust Algorithm

Equation (4-11) describes the relationship between the global reputation vector and the starting vector $\vec{l}_i$. According to the knowledge of linear algebra, if $L_{n\times n}$ is diagonalizable and its dominant eigenvalue is not equal to the second one, then a non-zero starting vector can converge to the dominant eigenvector after N

iterations. That is equivalent to saying that the non-zero starting vector does not affect the convergence of the global reputation vector. Therefore, a vector $\vec{n} = [1/n \quad 1/n \quad \cdots \quad 1/n]^T_{1\times n}$, where n is the amount of nodes in a given network, can be chosen as the starting vector in the simple centralized EigenTrust algorithm described in Figure 4-1 in which ε is a predefined error tolerance near zero.

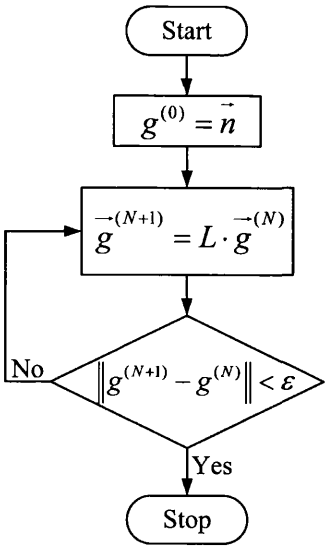

Figure 4-1 Simple EigenTrust Algorithm

Based on the algorithm description, a simulation of 10 nodes with 20 iterations was performed in Matlab. Figure 4-2 shows the change of the residual error (two-norm) of the vector $\vec{g}$ between before and after iterations. It can be seen from Figure 4-2 that the global reputation vector no longer changes dramatically after several iterations.

2) Improved EigenTrust Algorithm

As mentioned in Section 4.2.3, to increase the convergence rate of the simple EigenTrust algorithm, equation (4-12) was proposed. The improved EigenTrust algorithm is described in Figure 4-3. And a simulation of 10 nodes with 20 iterations was performed as well. Besides, there are three pre-trusted nodes in this simulation. The simulation result, as shown in Figure 4-4, proves the feasibility of the improved algorithm.

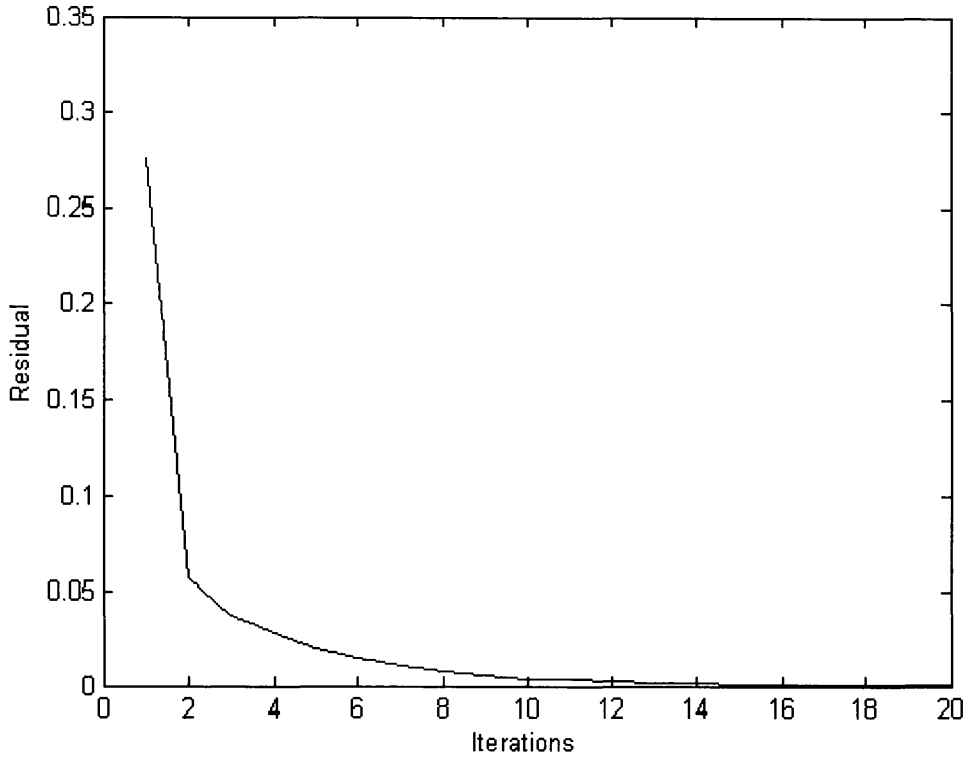

Figure 4-2 Simple Centralized EigenTrust Convergence ($\varepsilon = 0.001$)

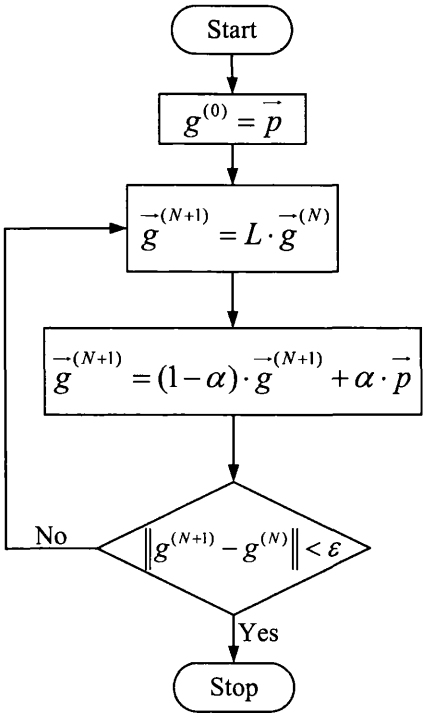

Figure 4-3 Improved EigenTrust Algorithm

Comparing the two simulation results, it is clear that the improved algorithm converges faster than the simple one. That is because equation (4-12) changes the

matrix L's second dominant eigenvalue λ_2 to the value of α, which has been proved in [35]. And because the basis of the EigenTrust is the power iteration algorithm whose convergence rate is determined by the ratio (λ_2/λ_1) between the second principle eigenvalue and the principle eigenvalue of the matrix[42]. Hence, the speed of convergence of the improved algorithm is α/λ_1. As long as the chosen α is greater than the original value of λ_2, the improved method can converge faster than the simple one. In our simulation, the original second dominant eigenvalue is 0.6332 which is less than the value of α that is set to 0.8. Thus, the decay rate of the residual in the improved EigenTrust is greater than that of the simple one.

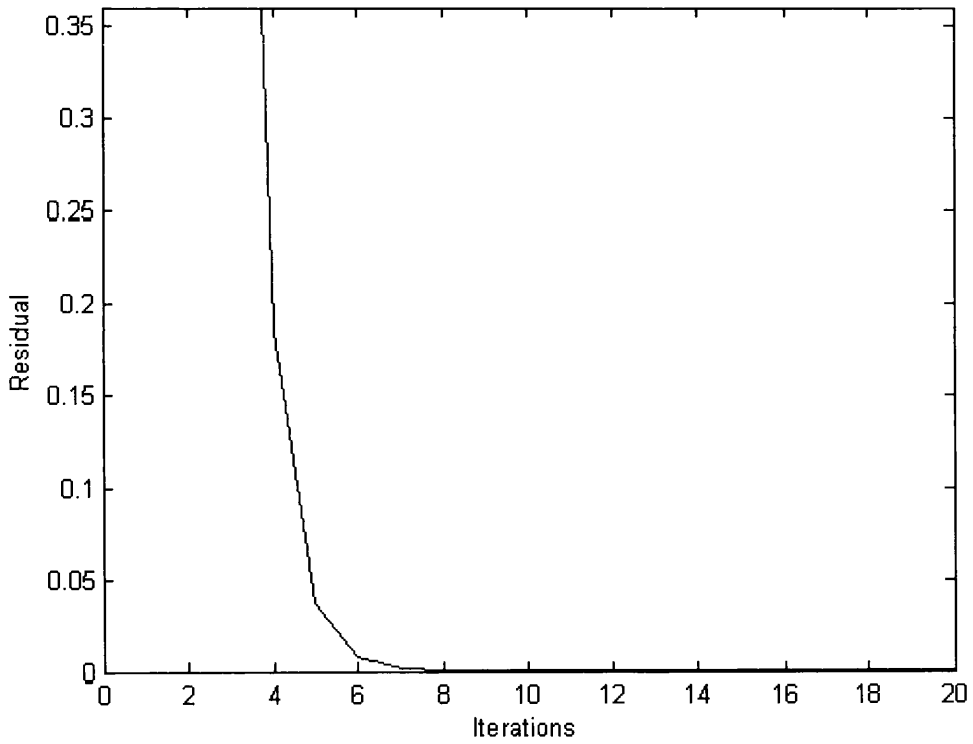

Figure 4-4 Improved EigenTrust Convergence ($P=3, \varepsilon=0.001, \alpha=0.8$)

4.3 Distributed EigenTrust Algorithm

An idea of this thesis is to apply EigenTrust to distinguish abnormal nodes from normal ones in WMNs. One feature of such networks is the self-organization that is the process where an entity appears without a central authority or an external intervention. And the organization appears a globally coherent pattern by local interaction among entities in a parallel (all the entities act at the same time) and

distributed (no entity is a central dictator) way[43].

Obviously, the centralized EigenTrust algorithms described in the previous section can not be applied in such distributed networks. Hence, it is necessary to present a distributed EigenTrust[4] where all nodes in a network cooperate to calculate the global reputation vector and each one stores one component of the vector.

4.3.1 Properties of Distributed Computation

A process of distributed computing is completed by a part or all of autonomous entities in a network, each of which has independent computing and storage units. A common computation task is subdivided into many small parts to such entities, and in general the output of each entity also interacts with others in the network to different degrees. In other words, every computational entity has its own co-operators whom the entity obtains data from or sends data to. So, at least, a distributed algorithm should endue each node with the following properties:

First of all, every node does not need to know the topology of the network, maybe composed by different kinds of entities, in advance. During the course of calculation, each node uses local and limited information from others to finish its own work. This does not mean that one's opinion is one-sided. For example, Section 4.3.2 shows that one entity can hold a wider view using an iterative way in the distributed network.

In addition, one's data should be available to other nodes by message passing, in order to promote computation efficiency and reduce resources costs. So, for another thing, the algorithm must be combined with a communication protocol for data sharing and hiding the heterogeneity of the network. Without such a protocol, a distributed method could not be implemented. In Chapter 5, the AODV routing protocol based on the distributed EigenTrust algorithm is designed and developed.

4.3.2 Algorithm Description of Distributed EigenTrust

An expression about the distributed EigenTrust algorithm can be derived from equation (4-12). For clarity, expanding this equation we have

$$\begin{bmatrix} g_1^{(k+1)} \\ \vdots \\ g_i^{(k+1)} \\ \vdots \\ g_n^{(k+1)} \end{bmatrix}_{n\times 1} = (1-\alpha)\begin{bmatrix} \bar{l}_{11} & \cdots & \bar{l}_{j1} & \cdots & \bar{l}_{n1} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \bar{l}_{1i} & \cdots & \bar{l}_{ji} & \cdots & \bar{l}_{ni} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \bar{l}_{1n} & \cdots & \bar{l}_{jn} & \cdots & \bar{l}_{nn} \end{bmatrix}_{n\times n} \begin{bmatrix} g_1^{(k)} \\ \vdots \\ g_i^{(k)} \\ \vdots \\ g_n^{(k)} \end{bmatrix}_{n\times 1} + \alpha \begin{bmatrix} p_1 \\ \vdots \\ p_i \\ \vdots \\ p_n \end{bmatrix}_{n\times 1} \tag{4-13}$$

Hence

$$g_i^{(k+1)} = (1-\alpha)\begin{bmatrix} \bar{l}_{1i} & \cdots & \bar{l}_{ji} & \cdots & \bar{l}_{ni} \end{bmatrix}_{1\times n} \begin{bmatrix} g_1^{(k)} \\ \vdots \\ g_i^{(k)} \\ \vdots \\ g_n^{(k)} \end{bmatrix}_{n\times 1} + \alpha \cdot p_i \tag{4-14}$$

Then

$$g_i^{(k+1)} = (1-\alpha)\cdot(\bar{l}_{1i}\cdot g_1^{(k)} + \bar{l}_{2i}\cdot g_2^{(k)} + \cdots + \bar{l}_{ji}\cdot g_j^{(k)} + \cdots + \bar{l}_{ni}\cdot g_n^{(k)}) + \alpha \cdot p_i \tag{4-15}$$

Notice that $g_i^{(k)}$ is not functional in equation (4-15). That is to say, both $g_i^{(k)}$ and $\bar{l}_{ii}$ do not take part into the process of computing i's global reputation. Because one's own opinion is probably too subjective to be true but the evaluation from others are more likely to be objective and accurate.

It should be noted that in equation (4-15) the unknown parameters are just neighbours' evaluations about node i and their own global reputation. (The values of α and p_i have been assigned before computation.) Moreover, even though there are $n-1$ unknown items in equation (4-15), the actual amount is small because only those nodes connecting with the node i have the local reputation about it; that is, equation (4-15) is sparse. This leads to two benefits. On one aspect, the computation load is lightweight for each node. On another aspect, the data to be transferred over the network is little as well.

If the node *i* has *m* neighbours, equation (4-15) can be written as

$$g_i^{(k+1)} = (1-\alpha)\cdot(\bar{l}_{1i}\cdot g_1^{(k)} + \bar{l}_{2i}\cdot g_2^{(k)} + \cdots + \bar{l}_{mi}\cdot g_m^{(k)}) + \alpha \cdot p_i \tag{4-16}$$

As shown in Figure 4-5, each neighbour only needs to send its up-to-date global reputation and local evaluation about the target node *i*. The rest calculations are

handed over to the target that are responsible for synthesizing the reputation data into its own global reputation as described in the equation (4-16). The synthesized reputation will be sent to its neighbours (e.g. node *j*) afterward for their own reputation calculation.

Figure 4-5 and equation (4-16) also show that *j*'s global reputation aggregates opinions of its neighbours including node *i*. Hence, $g_i^{(k+1)}$ also includes opinions from the other neighbours of *j*. Similarly, every neighbour of *j* takes further nodes' opinions into consideration. Thus, it can be deemed that *i* holds a global view over the entire network. Of course, it should be pointed out that one's neighbours' opinions are more weight for its global reputation than those distant nodes, because a distant neighbour's opinion will be repeatedly weighted by each of intermediate node's local reputation, less than 1, before its opinion is considered by the node *i*. This is similar to the social network. One is more likely to adopt friends' suggestions than those of unfamiliar persons.

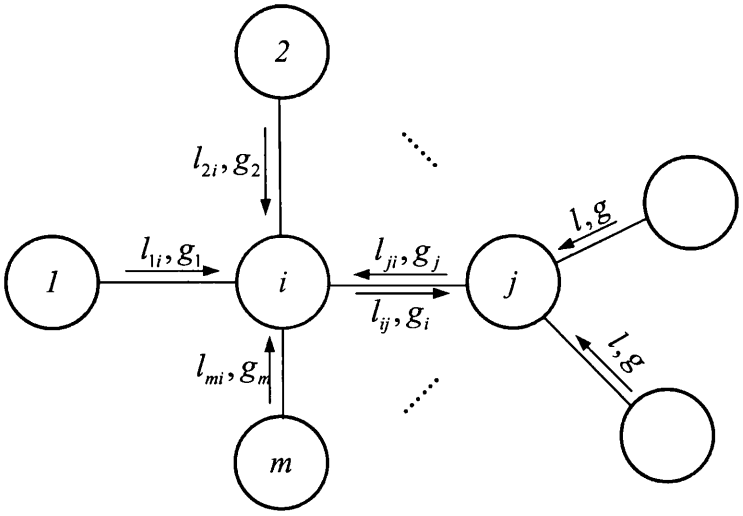

Figure 4-5 Aggregation of Reputation Data

Figure 4-6 describes the distributed EigenTrust algorithm. Particularly, node *i* initializes its global reputation ($g^{(0)}$) to p_i before calculating. The constants α and p_i have the same definition as we defined earlier. Then, if the node has received all the necessary data from neighbours, its global reputation can be computed based on these external data. After that, $g_i^{(k+1)}$ and $\bar{l}_{ij}$ are sent to *i*'s

neighbours that intend to compute their own global reputation. Finally, the residual error about global reputations in two adjacent iterations is computed and decides whether it is needed to do the next calculation loop or not.

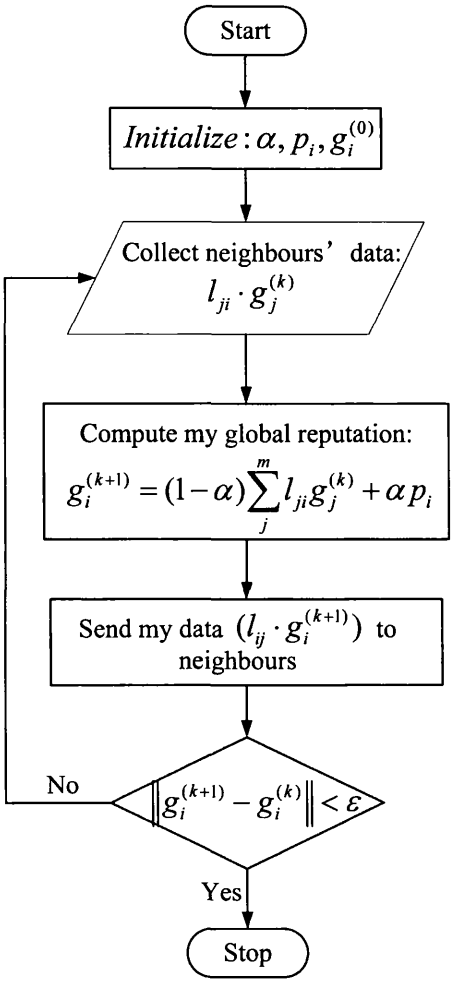

Figure 4-6 Description of Distributed EigenTrust Algorithm

4.4 Issues and Solutions for Implementation

When the distributed EigenTrust algorithm integrates with the AODV routing protocol, both of them need to be modified. With respect to the modification to AODV, Chapter 5 will illustrate this in detail. The following subsections are mainly about the alteration to the algorithm.

4.4.1 Redefining Local Reputation

According to the credit rating method in Section 4.2.1, the local reputation can be

computed as the difference between the number of packets forwarded by a given neighbour successfully and the number of packets forwarded unsuccessfully. But in our implementation, the local reputation of one's neighbour is redefined as the proportion of packets forwarded by the given neighbour in overall packets forwarded by all its neighbours.

4.4.2 Initial Values for Local and Global Reputations

The initial value of the local and the global reputation need to be carefully selected in the EigenTrust algorithm. The initial value of local reputation towards a given node is set to 0.01 because at the start-up phase packets have not been forwarded by the neighbour. The reason why the initial value takes 0.01 rather than zero will be explained in Section 4.4.3.

With respect to the normalization, local reputations about one's m neighbours is rewritten as equation (4-17). If all the m neighbours have not forwarded any packet, their normalized local reputation will be a uniform probability distribution over them, namely $[1/m, 1/m, \cdots, 1/m]_{1\times m}$.

$$\bar{l}_{ij} = \frac{l_{ij}}{\sum_{\substack{j=1, \\ j\neq i}}^{m} l_{ij}} \tag{4-17}$$

An assumption in our implementation is that every node should have been friendly, so one's initial global reputation is set to 1. And the initial global reputation must not take zero, because a non-zero starting vector $\vec{g}^{(0)}$ is required in the EigenTrust algorithm. After several interactions with other nodes, a node can figure out a new global reputation that may be lower than its initial value.

4.4.3 Pre-trusted Nodes

As mentioned in Section 4.2.3, the existence of pre-trusted nodes can ensure the convergence of equation (4-15). In practice, the selected pre-trusted nodes should not belong to a set of malicious nodes such that the performance of EigenTrust would not be compromised. So the authors in [4] suggested that the first few nodes joining a

network could be regarded as trustworthy and these nodes should be selected as few as possible.

However, the selection method is still an open research area[3] and improper selection also compromises the convergence. Moreover, there is no method to count the number of the pre-trusted nodes in AODV. So a practical challenge is to guarantee the distributed EigenTrust convergent when no pre-trusted nodes are specified. The following will discuss the convergence of equation (4-18) in that situation.

$$g_i^{(k+1)} = (\bar{l}_{1i} \cdot g_1^{(k)} + \bar{l}_{2i} \cdot g_2^{(k)} + \cdots + \bar{l}_{mi} \cdot g_m^{(k)}) \tag{4-18}$$

The irreducibility and aperiodicity of $L_{n\times n}$, which are the convergence premise of EigenTrust[4], cannot always be achieved without the pre-trusted nodes. To overcome this limitation, the distributed EigenTrust is correspondingly modified in terms of the initial value and termination condition.

First, we consider the irreducibility. According to the definition of the irreducible matrix, an irreducible matrix has at least one non-zero off-diagonal element in each row and column[44]. Hence, if there is no zero element in the subdiagonal and superdiagonal of $L_{n\times n}$, the irreducibility of the matrix can be warranted.

Take Figure 4-7, where '×' represents a random value of the local reputation, as an example. It can be seen that the elements of the superdiagonal of $L_{n\times n}$ are created during the routing discovery of AODV and the superdiagonal values are set when the RREP is unicasting back to the RREQ source node. Thus, as long as the nodes in a route set a non-zero local reputation to their downstream and upstream neighbours, the irreducibility of $L_{n\times n}$ can be guaranteed. Therefore, the initial value of local reputation takes a decimal value near zero (e.g., 0.01) in equation (5-1).

$$l_{ij} = \begin{cases} \dfrac{pkts_j}{pkts_{tot}} & if \quad pkts_j \neq 0; \\ 0.01 & otherwise \end{cases} \tag{4-19}$$

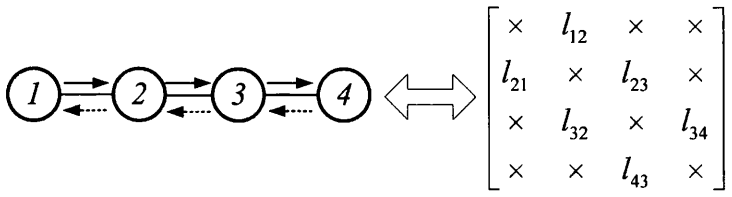

Figure 4-7 An Irreducible Matrix

On another aspect, it is assumed that the matrix $L_{n\times n}$ is said to have period k, namely $L^{k+1} = L$ where k is the least such positive integer. Hence we have $\vec{g}^{(k+1)} = L^{k+1} \cdot \vec{g}^{(0)} = L \cdot \vec{g}^{(0)} = \vec{g}^{(1)}$. According to the simulation results in Figure 4-2 and Figure 4-3, the residual error $\left\| \vec{g}^{(k+1)} - \vec{g}^{(k)} \right\|_2$ in the centralized algorithm monotonically decreases over iterations. Similarly, for the distributed method, the residual error $\delta = \left\| g^{(k+1)} - g^{(k)} \right\|_2$ should monotonically decrease over iterations as well. Obviously, considering the periodicity of the matrix, at the $(k+1)^{th}$ iteration the error $\delta^{(k+1)} = \left\| g^{(k+1)} - g^{(k)} \right\|_2$ can be written as $\delta^{(k+1)} = \left\| g^{(1)} - g^{(k)} \right\|_2$. Due to the decrease of δ, $\left\| g^{(1)} - g^{(k)} \right\|_2$ will be much larger than $\delta^{(k)}$. Therefore, we can use this significant change of δ as a subsidiary condition for deciding whether to continue the calculation or not. In other words, if $\delta^{(k+1)} \geq \delta^{(k)}$, the calculation process has to be stopped and waits for being restarted later; otherwise, it keeps going on. Before the $(k+1)^{th}$ iteration, the calculation can be considered as convergent.

Based on the above discussion, through modifying the initial value of local reputation and adding the subsidiary termination condition, the convergence of the equation (4-18) can be guaranteed. The convergence is also proven in the experiment illustrated in Chapter 5.

4.5 Summary

The accuracy of intrusion detection is a challenge for an intrusion detection system. Although multi-point detection can significantly improve the accuracy compared with the single-source detection, the application of multi-source detection is limited

by the reputation calculation methods. Therefore, EigenTrust is introduced to evaluate the consensus reputation for every node in a network. Because the convergence of EigenTrust would be compromised when no pre-trusted nodes are defined and there is no such nodes exist in AODV, the algorithm is improved to guarantee the convergence at the cost of reducing convergence rate when no trusted node is pre-defined. The corresponding implementation in AODV will be shown in the next chapter.

5 A Reputation Based Secure AODV for WMNs

To overcome the packet drop attack, a secure AODV is proposed and implemented in this chapter for intrusion detection in WMNs. The protocol is named AODV-ET whose reputation algorithm has been presented in Chapter 4. The following sections describe the protocol in detail.

5.1 Modification of RREQ

As described in Section 4.4, the RREQ routing message should include reputation information in order to apply EigenTrust into AODV. Therefore, there is necessary to modify the format of the standard RREQ. As shown in Figure 5-1, there are two places highlighted in solid boxes different from a standard RREQ message.

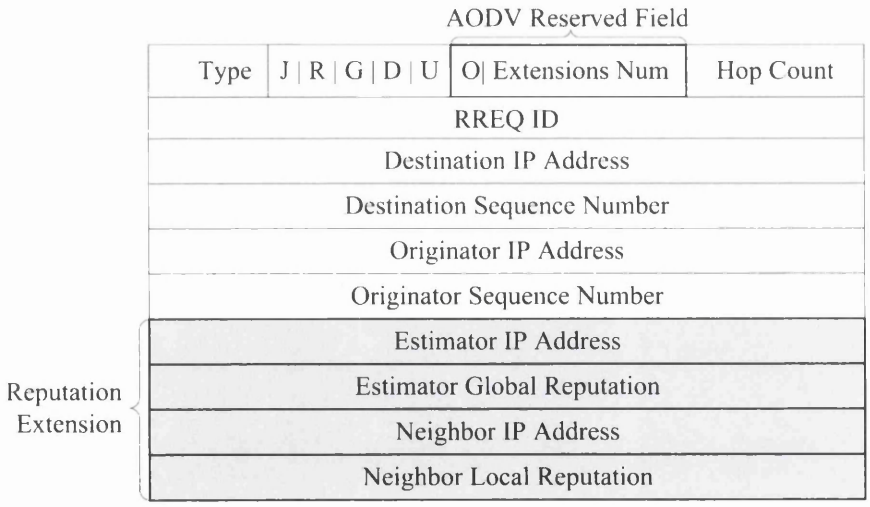

Figure 5-1 Message Format of RREQ with Reputation Extension

Like AODV-REX[32], the first modified is the reserved field of the standard RREQ which is 11 bits length, set as 0 and ignored on reception. That is to say, the field is not used by AODV, so modifying it will not affect the process of RREQ operation in AODV. The value of the first bit, named 'O', of this field represents whether a RREQ message with reputation extensions or not. If this bit is set to 1, the RREQ's receiver will know that there are reputation extensions attached. Moreover, the rest 10 bits of the field stand for the number of reputation extensions.

Besides, reputation extensions are added at the end of the standard RREQ message. One extension includes four fields: Estimator IP Address, Estimator Global Reputation, Neighbour IP Address, and Neighbour Local Reputation. The first field stores the IP address of a rater. The following keeps global reputation of the rater. The next two saves the ratee's IP address and the evaluation that the rater puts on the ratee, respectively. If a rater has more than one neighbour (ratee), their corresponding reputation extensions will be one by one added at the end of the RREQ which the rater creates and forwards.

5.2 Reputation Table

At every node, a structure variable named Reputation Table (RT) is defined as below.

```
typedef struct list_t {
    struct list_t *prev, *next;
} list_t;


struct Reputation_Table {
   struct in_addr nghb_addr;   /* IP address of a neighbour */
   float reputation; /* Global reputation of the neighbour*/
   int nghb_pkt_fwd; /* Number of packets forwarded to a neighbour */
   list_t lnk_RT;              /* Link about RT */
};
```

A RT is used to encapsulate reputation information of one's neighbour: the neighbour's IP address and global reputation, as well as the number of packets forwarded by the neighbour. A node creates a RT for each neighbour and every RT is linked as a sequence by a list structure (*lnk_RT*) that is efficient to insert a new RT anywhere and iterate over in forward or reverse order.

Notice that local reputation value about a neighbour is not included in this structure, because in fact the member *nghb_pkt_fwd*, recording the number of packets that the neighbour has forwarded, is corresponding to the neighbour's local reputation. This corresponding relation is shown in the equation (5-1). Besides, one node's own global reputation value is separately saved in a pre-declared global variable.

The creation of the RT can be done in the following two ways. If a node receives a RREQ from a neighbour, a RT will be created for the neighbour by the node. Likewise, when the node receives an unseen RREP from a neighbour, a new RT about this neighbour will also be established. In these ways, a node creates a RT for each neighbour.

5.3 Propagation of RREQ

As mentioned in Section 4.3.1, the AODV protocol is used to transfer required data for the EigenTrust algorithm. Specifically, before one node broadcasts a RREQ message, its global reputation and neighbours' local reputations are computed and added into the message. When an intermediate node receives the RREQ, it computes its own global reputation using the corresponding data in extensions of the RREQ. Additionally, the intermediate node attaches its own reputation data to the RREQ and forwards the RREQ if needed. With the propagation of RREQs, each node's reputation data spreads over a network. The detailed explanation is as following.

5.3.1 Creation

According to the modified format of RREQ, the process of generating a RREQ for discovering a path to a destination in AODV-ET is described in Figure 5-2.

First, a node originates a standard RREQ message including flags, address information, sequence numbers and so on. Next, the first bit of reserved field of the message is set to 1 so that a receiver can recognise prospective reputation extensions in the RREQ. Then the IP address and global reputation of the node (rater), as well as the IP address, local reputation of each neighbour (ratee), are stored in a structure array *extRep*. After that, the number of reputation extensions and the size of the RREQ with the extension are updated so that a receiver can read reputation data in the RREQ and add new reputation extensions at correct place. Moreover, all members of the structure array are packed and placed in the rear of the RREQ one by one. Finally, the RREQ with extensions is sent to a Linux Kernel socket for broadcasting.

```
Definitions:
-k: counter of extensions
-extRep: a structure array storing reputation data

Process:
Create a standard RREQ;
Set flag bit of reputation extension;
foreach neighbour j
{
    extRep[k].rator_addr= localhost IP address;
    extRep[k].reputation= global reputation;
    extRep[k].ratee_addr= j's IP address;
    extRep[k].local_reputation= j's local reputation;
    k++;
}
Update: size of the RREQ with extensions, the number of extensions;
Add elements of extRep[] to end of the RREQ;
Broadcast the RREQ wtih extensions on socket;
```

Figure 5-2 RREQ Creation

5.3.2 Reception

Due to the modification to RREQ, the reception process of a RREQ in AODV-ET is slightly different from the AODV protocol. The difference is highlighted in Figure 5-3.

The first difference is that a RREQ source node needs to receive any RREQ even if those are created by itself and then passed back from its neighbours. This is because the source needs to extract the reputation data about its neighbours from the RREQ in order to calculate its own global reputation. After picking out the needed information, the node drops the RREQ generated by itself. By contrast, in AODV a node should ignore its own RREQs at once when receiving them.

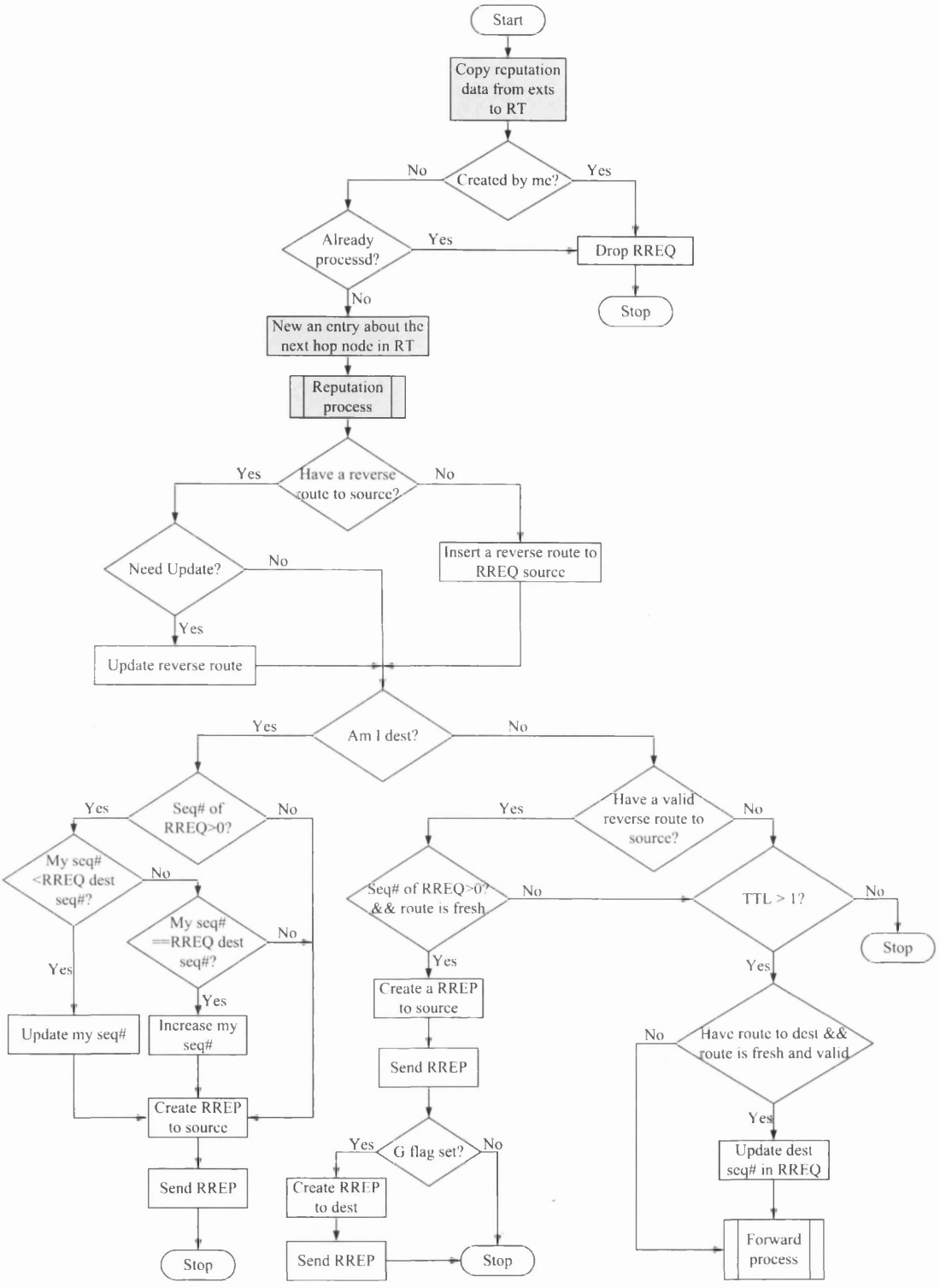

Figure 5-3 Flowchart of Processing RREQs

Additionally, when receiving a new RREQ during route discovery, a node creates a new RT about its previous hop (upstream neighbour) before the RREQ is processed. Then the reputation data included in each RREQ extension is used to update the corresponding RT's content if such fields can be found. Finally, the reputation

calculation process, discussed in Section 5.4, is called to compute the current node's global reputation according to the reputation data placed in RREQ extensions. Contrary to AODV-ET, for the new unprocessed RREQ, AODV will not do the above work but check local reverse routing table to decide whether to update the reverse route or to insert a new one to the RREQ source node.

If anything, a minor difference is to create a RREP for responding to a RREQ source node, which is shown in the bottom of Figure 5-3. The value of hop count in RREP is virtual hops rather than a real physical hop. Actually, the virtual hop count is the reputation metric described in Section 5.5.1.

5.3.3 Forwarding

If a node does not have a route to the destination of a received RREQ, it has one but the route has been invalid, or the route is not fresh enough, then the node will start a process forwarding the received RREQ.

Figure 5-4 shows this process. To begin with, the coming RREQ should be stored in a sending buffer so that prospective reputation extensions could attach to this RREQ, and its hop count should also be increased by 1. Then the current node checks its RTs to judge whether it owns such information that is not included in the extensions of the RREQ or not. If any, the data would be packed as a new extension and added to the end of the RREQ; otherwise, reputation information in extensions would be updated by its counterpart in RTs so that data in RREQ extensions is up-to-date. Finally, the RREQ with extensions is sent to the Linux Kernel socket for broadcasting to neighbours.

5.4 Reputation Calculation Process

5.4.1 Local Reputation Calculation

A neighbour's local reputation is based on its behaviour history detected by the two modules explained in Section 5.6. Here we only concentrate on its calculation. Assume that the node j is one of the node i's neighbours. The node i is able to sense the number of packets forwarded by j ($pkts_j$) and the number of packets

sent out by all neighbours ($pkts_{tot}$), as shown in Figure 5-7. According to Section 4.4.1 and 4.4.2, local reputation l_{ij} is computed as follows:

$$l_{ij} = \begin{cases} \dfrac{pkts_j}{pkts_{tot}} & if \quad pkts_j \neq 0; \\ 0.01 & otherwise \end{cases} \tag{5-1}$$

Notice that if there is no packet forwarded by all the neighbours, which means the connection has just been established, the initial value of local reputation of the neighbour j is set to 0.01. Besides, before the local reputation is used in computing one's global reputation, it must be normalized by equation (4-17).

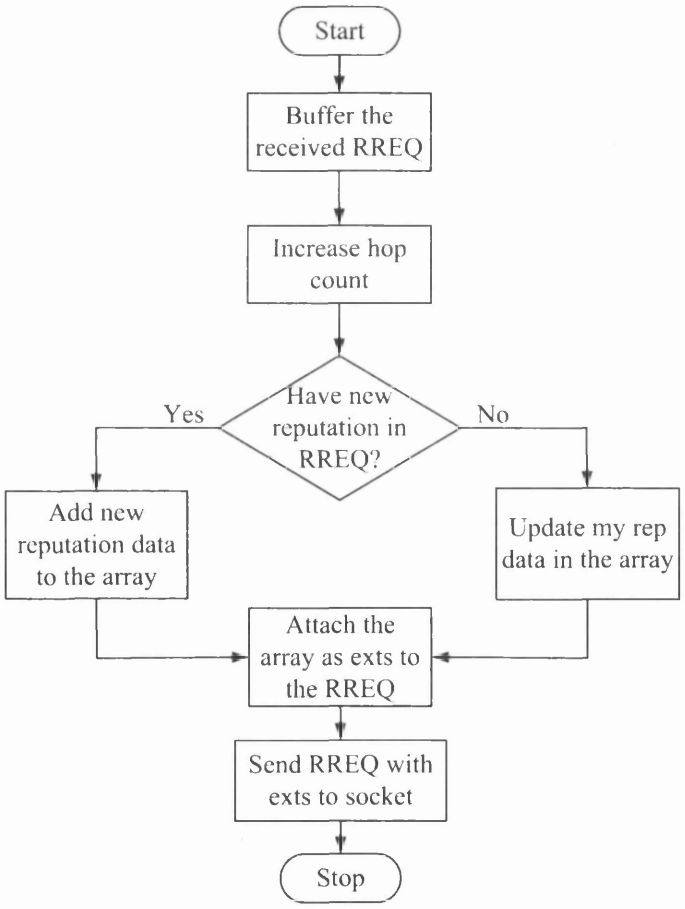

Figure 5-4 Flowchart of Forwarding RREQs

5.4.2 Global Reputation Calculation

The process of computing the global reputation is a part of processing a received RREQ in AODV-ET. As highlighted in Figure 5-3, the function *reputation_process* is

called before one node updates or adds a reverse route. Particularly, Figure 5-5 shows the process of calculating one's global reputation in detail.

For a received RREQ with reputation extensions, a node needs to find out its neighbours' global reputations and local reputations about itself. To pick out the information, the first thing the node should do is to check if the Ratee IP Address of an extension matches the current node's IP address and the rater is its neighbour or not. If a rater's IP address can be found in a ratee's RT, it is deemed that the rater is a neighbour of the ratee. Because one's RT only stores its neighbour's reputation data.

Finally, the result of each loop is accumulated to get a global reputation as shown in Figure 5-5 and equation (4-18). Besides, one should also update its rater's global reputation in its RTs with the corresponding data in RREQ extensions so that the reputation metric is accurately computed in Section 0.

5.5 Route Selection

In general, establishment of a routing table in a routing protocol is subject to a link cost factor (external metric) associated with other entities in a network. Cost factors may be the distance to a destination, network throughput of a link, or link availability and reliability, expressed as simple unitless numbers[45]. As presented in Section 2.3.2, in AODV the hop count from a source to a destination is such a cost factor for selecting the shortest path. This provides a dynamic process of traffic load balancing between paths.

Unlike the AODV routing protocol, the establishment of a route entry in AODV-ET is governed by a combined factor: reputation metric.

5.5.1 Reputation Metric

The reason of introducing EigenTrust into the AODV-ET routing protocol is to assign the cost factor in AODV a property indicating the degree of reliability of a route for avoiding malicious nodes. For this purpose, the reputation metric, also named reputation hops, is used to represent virtual hops taking global reputation level and a real physical hop between two adjacent nodes into consideration.

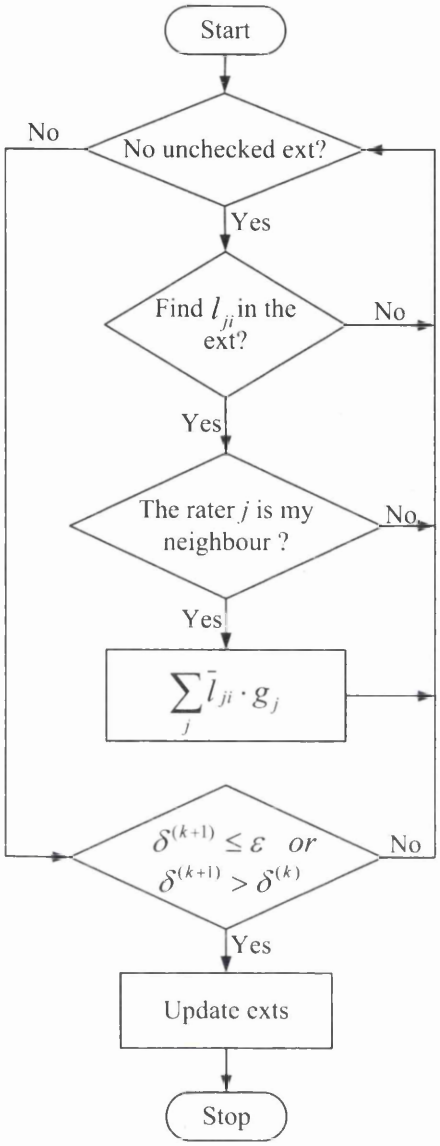

Figure 5-5 Process of Global Reputation Calculation

The metric is negative related to a node's the global reputation value of each. In other words, a low global reputation leads to a long distance, while a higher one results in fewer hops. As shown in formula (5-2), the global reputation of the downstream neighbour j, $g_j \in [0,1]$, is converted to the reputation hop.

$$RH_{ij} = round((1 - g_j) \cdot ND) + 1 \tag{5-2}$$

where, reputation metric RH_{ij}, may not equal to RH_{ji}, denotes the hop count from i to j. It takes j's global reputation and the normal hop (constant 1) into account. Clearly, if j enjoys high enough reputation, RH_{ij} tends to the normal hop 1.

Moreover, *ND* stands for the Network Diameter that specifies the maximum possible number of hops between two nodes in a network. Besides, a round-up function is used to map a float number to an integer rather than the floor function applied in [32]. Because the round-up function can more obviously reflect the change of total hops of a route.

5.5.2 Most Trusted Path First

The creation of a new route is initiated by sending a RREQ and receiving a corresponding RREP message[33]. If several valid routes towards a destination exist, the destination can receive several RREQs. In basic AODV, a source selects the route with minimum number of hops to an intended destination, as explained in Section 2.3.2.

Similarly, AODV-ET selects the most trustful route to a destination based on SPF (Short Path First). In order to select a reliable route, the destination in AODV-ET needs to answer every RREQ message that is forwarded towards it via different paths, rather than only reply the first coming RREQ like AODV. Two options can be considered for this situation. Either the choice of a path is decided by the RREQ source and the destination will be notified (source-oriented), or, conversely, the destination makes the decision and informs the source (destination-oriented) [33].

According to [32, 33], the source-oriented approach is illustrated in Figure 5-6. It can be generalized to three steps. Firstly, the source node S broadcasts a RREQ. Secondly, the destination D receives several RREQs from different paths, and responds to each of them by a RREP which returns along the path which the RREQ traversed before. Finally, S receives several RREPs, each of which contains the metric representing reputation level of the path they walked through. And then S chooses the most trustworthy route and, if requested, informs the destination by an acknowledge message.

In other words, each node in a route, except the source node, adds the reputation metric to Hop Count of a RREP to indicate a virtual distance to the destination that created the RREP. This distance takes the reputation of every intermediate node into

account. When the source receives the RREP, it is able to recognise the trustworthiness of the route to the intended destination by the value of Hop Count in the RREP.

Alternatively, the destination-oriented approach would proceed as follows. First of all, S sends a RREQ by broadcast. When receiving the RREQ, intermediate nodes update its hop count (reputation metric) value and forward the updated RREQ. At last, the destination receives several RREQs including reputation hops of each path. It chooses the most reliable route and only sends a single RREP to the source along the route it selected. After received the RREP, the source updates or inserts a route to the destination in its routing table.

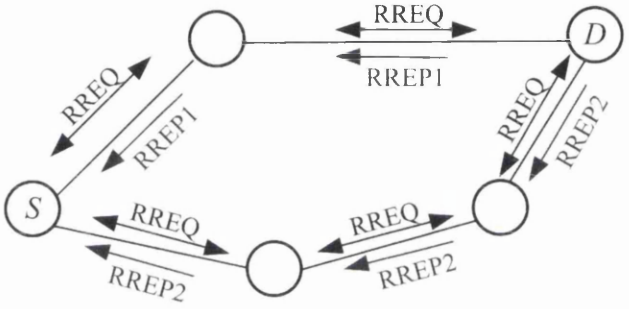

Figure 5-6 Source-oriented Approach

Obviously, the two approaches are different. In the first method, the reputation metric is computed when receiving and replying a RREQ. But the destination oriented way only performs calculation and update of the metric during the propagation of RREQ. In addition, the source node is responsible for selecting a trustworthy route based on received RREPs. In the destination oriented way, conversely, the destination node chooses the reliable route before sending a single reply so that the source can directly create or update a route when receiving the reply.

Hence, the destination oriented approach selects a reliable route quicker than its counterpart because the source node does not need to spend time comparing reputation metrics of all received RREPs. Remarkably, its inherited weakness is that the destination node must be trustful. If the destination replies a RREP with false reputation metric to attract network traffic, performance of the destination-oriented method would be reduced. However, the source oriented approach can avoid this

issue because the reputation metric of a RREP can only be modified by the upstream node in a reverse route to the source. If unfortunately, a downstream node is malicious, its upstream neighbour would give it a low reputation leading to high hops so that the path through the malicious would not be selected. Therefore, the source-oriented approach is used in AODV-ET.

To validate AODV-ET, routing behaviour of packets should be monitored. Details about auditing routing of data packets are illustrated in the following.

5.6 Misbehaviour Detection

Network Sniffer and Watchdog can be employed to monitor the behaviour of a node's neighbours. The former takes in charge of capturing and analysing data packets. The latter judges whether a packet is successfully forwarded. They are, respectively, described in detail as following.

5.6.1 Sniffing Principle

A Sniffer is a piece of computer programme that can grab all or part of traffic passing over a network. The network sniffing technique can be applied to detect misbehaviour of an entity in a network, e.g., dropping packets. This is possible since the wireless channel is a broadcast medium; if the reception and transmission ranges are the same, every forwarding activity of a node can be sensed by all its neighbours[32], and vice versa.

Certainly, the application of sniffing technique depends on the mode of NIC (Network Interface Card). In a real system, a NIC is responsible for operation of packets, whose main job is to collect packets traversing through it. When the NIC receives a packet, it will decide whether to accept it or not based on its mode and the destination's information included in the packet. The packets whose information is not on the need will be ignored. Generally speaking, there are four modes of NIC:

(1) Broadcast: A NIC can receive broadcast messages over a network.

(2) Multicast: A NIC can receive messages from the same group with it. That is, a packet can be received by more than one NICs belonging to the same group. They

communicate with each other by a broadcast-style.

(3) Directory: A packet is only permitted to be received by the intended NIC of destination.

(4) Promiscuous: A NIC is able to receive any data passing through it.

By default, the NIC supports broadcast and directory mode, namely it can receive broadcast messages and the messages to itself. Specially, if a NIC intends to capture packets in a network, the promiscuous mode must be supported.

The application of sniffing technique, which is called network sniffer and based on *Socket*, *Libpcap* or *WinPcap*, enables a NIC to capture all of the network packets in real-time passing through it. The sniffer, based on the *Socket*, only can respond to a broadcast packet whose destination MAC address is matching with its NIC's MAC address. For other packets, it is unable to do anything.

However, the network sniffer using *pcap* (*packet capture*), consisting of API (Application Programming Interface) functions, is able to capture all packets travelling through the local NIC. Therefore, the *pcap* is used by the sniffer in our implementation.

5.6.2 Network Sniffer Programming

The *libpcap* library is the *pcap*'s Linux application, and its Windows counterpart is known as *WinPcap*. In AODV-ET, *Libpcap* is used to sniff network traffic and the design process[46] of corresponding sniffer is described as below.

1) Setting the device

First, it is essential to declare the name of the interface which we want to sniff on, such as eth0, wlan0, etc. There are two ways to do that. One is to get the interface name by a passed string variable or a character pointer, like:

```
char *dev_name = "wlan0";
```

The other is to use a function, *pcap_lookupnet()*, to return the interface name, e.g.:

```
char *dev_name = pcap_lookupdev(errbuf);
```

The string variable *errbuf* stores an error description if the function *pcap_lookupdev* fails.

For simplicity, in AODV-ET the first method is used to set the device which we wish to sniff on.

2) Opening for sniffing

Then, the next step is using the function *pcap_open_live* for opening the device to sniff, like:

```
pcap_t *handle = pcap_open_live(char *dev_name, int snaplen, int promisc,
int to_ms, char *errbuf);
```

Because *libpcap* supports listening to multiple interfaces, this function returns a session handle to differentiate the specific interface being sniffed on from other interfaces. In particular, the first argument indicates the interface which we want to sniff on. The *snaplen* is an integer that defines the maximum number of bytes to be captured. The third one *promisc*, when set to true, brings the interface into promiscuous mode, not vice versa. The following argument, *to_ms*, is the read time out in milliseconds, which means when the function works after a sufficient number of packets arriving. So a non-zero integer, e.g. 500, should be assigned to that argument. As mentioned above, the last *errbuf* is a string variable to store error messages. Finally, the function returns a session handler.

3) Set a traffic filter

After that, it is necessary to select out the specific type of packets (e.g. tcp, icmp or udp), which we are interested in, from all received packets. That is, filtering traffic. The filter of *pcap* is based on BPF (Berkeley Packet Filter[47], provide a raw interface to allow receiving packets destined to other hosts) that is more efficient and easier than users own *if/else* statements. Two API functions in *libpcap*, as below, can be used to complete the filtering task together.

```
/* compile the filter expression */
int pcap_compile(pcap_t *handle, struct bpf_program *fp, char *str, int
```

```
optimize, bpf_u_int32 netmask);
 /* apply the compiled filter */
 int pcap_setfilter(pcap_t *handle, struct bpf_program *fp);
```

The function *pcap_compile* compiles a filter expression before we apply BPF. It returns a negative integer, -1, on failure, and other values indicate success. Specifically, it is necessary to tell *libpcap* that which interface should be filtered so the first argument (session handle) is used to identify an interface. Following that is a reference to the place that stores the compiled version of the filter. Then comes the filter expression (e.g. *icmp*) in string format. Next one is an integer that decides whether the expression should be 'optimized' or not. Finally, the net mask of the network which the filter applies to must be specified.

After the filter expression has been compiled, the second function *pcap_setfilter* can apply the expression by the first two arguments to *pcap_compile*, namely the session handle and filter version.

4) The actual sniffing

Finally, the function *pcap_loop* starts a loop to capture packets. The prototype of this function is as follows.

```
 int pcap_loop(pcap_t *handle, int num_pkts, pcap_handler got_packet,
u_char *user);
```

It will continue capturing packets until the count of packets (*num_pkts*) runs out. If the value of *num_pkts* is set to -1, the capture process will keep going on unless the application is forced to close. The preference *got_packet* is a handle of callback function that is an interactive interface with other applications, e.g., the watchdog module. The last argument is generally set as NULL.

Users can write some statements in the callback function *got_packet* to complete a specific task, such as dissecting a packet and print it to the user, saving it in a file, or doing nothing at all. In AODV-ET, the callback function determines whether a captured packet created by the local host have been forwarded by a given neighbour or not. Also it calls a function, *process_fwd_packet*, defined in watchdog module to judge whether a captured packet created by the local host has been forwarded by a

given neighbour. If it does, the reputation of the given neighbour will increase. After sniffing work is finished, similar to a file operation, it still needs to clean up memory using the below two functions:

```
void pcap_freecode(&fp);
void pcap_close(handle);
```

5.6.3 Watchdog Module

The watchdog module of AODV-ET is a piece of programme judging whether one node's neighbour forwards data packets successfully. It aims to evaluate local reputation level about a neighbour based on observation of the sniffer mentioned above.

Specifically, when a data packet is sent to a neighbour (intermediate node), the watchdog stores information about the packet in a buffer and immediately activates a timer. If within timeout the sniffer senses a sent data packet whose information matches that one buffered before, the watchdog thinks that the data packet has been successfully forwarded to the given neighbour and increases the value of *nghb_pk_fwd* in the RT corresponding to the neighbour, and deletes the buffered information; otherwise, if the timeout happens, the watchdog will not increase the value of *nghb_pk_fwd.*

In order to estimate local reputation, a queue is used to record the observation results in [32] for considering past interactions between a node and its neighbour. In AODV-ET, however, the observation queue is discarded because the ratio between the number of data packets forwarded by a given neighbour and the total number of data packets forwarded by all neighbours is used to represent the local reputation about the given neighbour. The ratio also reflects a history of the neighbour's behaviour in the past.

Of course, it indeed cannot avoid sporadic interference that is not caused by malicious actives. But this is insignificant. First of all, a small number of packets delivery failures, attributed to link failures, affect a neighbour's local reputation slightly, compared with the intentional and periodical misbehaviour. Moreover, even

if the failure of link to a neighbour periodically happens in certain situation where there is no malicious node, the neighbour can also be marked with a low reputation to avoid being selected. So, that ratio not only indicates the evaluation to a neighbour but also availability and reliability of the link connecting to the node.

5.6.4 Cooperation between Watchdog and Sniffer

Figure 5-7 shows the relationship between the watchdog and sniffer modules.

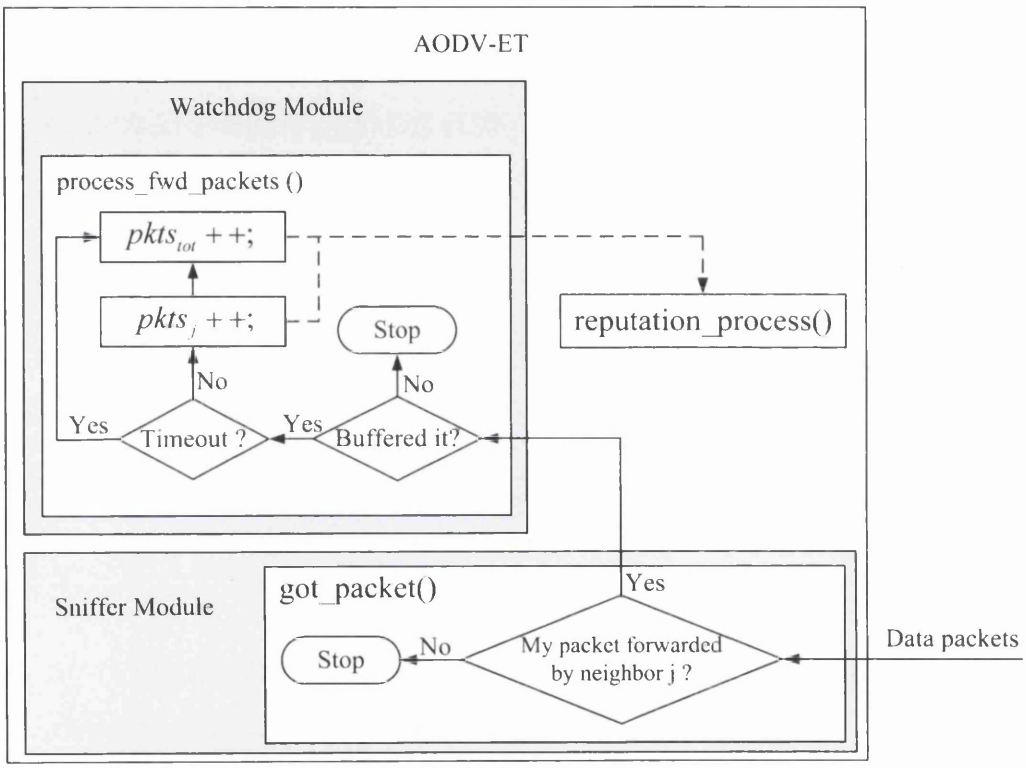

Figure 5-7 Relationship between Watchdog and Sniffer

The sniffer module takes in charge of capturing data packets going through the local host. It uses the function *pcap_loop* to call the function *got_packet* for capturing and analysing a packet. By checking the IP header of packets, the sniffer can judge if the host's packet were forwarded by neighbours successfully. And then, the *got_packet* function calls the function *process_fwd_pkt* of watchdog module to decide whether the captured packet is forwarded within a specific time. If a timeout occurs, the packet is deemed to be forwarded unsuccessfully by the neighbour. In this case, node i only increase the value of $pkts_{tot}$. Otherwise, both $pkts_j$ and $pkts_{tot}$ plus 1.

The updated $pkts_j$ and $pkts_{tot}$ will be employed in calculating a node's global reputation, if such a process, *reputation_process*, is triggered by AODV-ET.

5.7 Experiment and Results

In the following sections, the protocol AODV-ET will be tested in a real indoor environment. First, descriptions of devices condition and setup for the test infrastructures are given out. Then the approach used to test this routing protocol is described. Finally, experimental results are presented.

5.7.1 Testbed Setup

1) Hardware Configuration

The testbed for AODV-ET consists of 4 computers: 1 laptop and 3 Dell desktops (Figure 5-8). Their configuration can be seen from Table 5-1. Because the wireless card is not embedded in this type of desktops, it is necessary to use the external wireless adaptor to perform wireless connection. In our experiment, the DWL-122 adaptor of D-Link, compatible with 802.11n/b/g devices and supporting promiscuous mode, is employed. Of course, the wireless card of the laptop supports this listen mode as well.

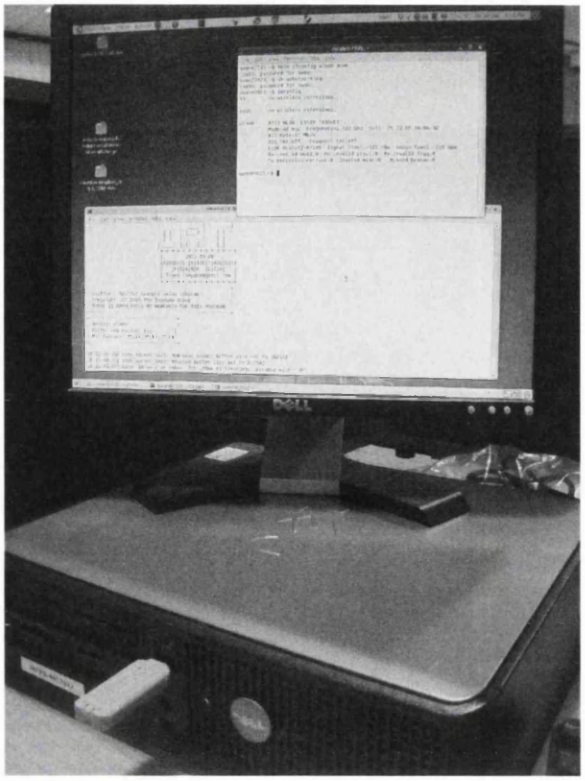

Figure 5-8 One Node of Testbed

Table 5-1 Testbed Configuration List

Type	Desktops	Laptop
Brand	Dell GX520	Lenovo G430
CPU	Intel Pentium 4 Processor 630	Intel Dual Core T4200
RAM	1G	2G
Hard Disk	160G	250G
WLAN Card	DWL-122 USB wireless adaptor	Broadcom BCM4312

2) Software Requirements

Table 5-2 lists the basic information about essential software for AODV-ET.

Table 5-2 Software List

Type	Name	Version
Operating System	Ubuntu	8.0.4.4
Linux Kernel	Kernel	2.6.24-26-generic
WLAN Card Driver	RT73;	2009041204;
	STA	5.100.82.112
Packet Capture Tool	Libpcap0.8;	0.9.8-2;
	Libpcap0.8-dev	0.9.8-2
Compiler	GCC	4.2.4

Ubuntu, a Linux distribution, is selected as the operating system of the testbed. The most important reason is that we developed AODV-ET on the basis of AODV-FUUREX[33] that is based on AODV-UU[48, 49] firstly published by Uppsala University in Sweden. They both run as a user space daemon with Linux kernel 2.6.x.

Moreover, our test infrastructures can meet the minimum requirements[50] of Ubuntu desktop edition, such as 1 GHz CPU (x86 processor, Pentium 4 or better), 1 GB RAM (system memory) and 15 GB of hard-drive space.

Furthermore, compared with Debian, Ubuntu compiles its packages using GCC[51] features, such as buffer overflow protection. Such extra features greatly increase its security. Besides, Ubuntu based on the Debian Linux distribution is distributed as free and open source software.

For the above reasons, eventually, Ubuntu 8.04.4 whose kernel version is 2.6.24-26-generic is installed on all testing machines.

3) Topology Setting

In order to test AODV-ET, the topology of the Ad hoc network is set as shown in Figure 5-10. But, since we do not have a room large enough to deploy the four

computers such that two neighbouring nodes are the only nodes within the transmission range of a given node, it is necessary to do some special setup to prevent the given node directly communicating to every possible destination node.

A built-in Linux tool *iptables* allowing dropping packets based on IP or MAC address is used to achieve that. The *iptables* is a user space command line program for configuring the Linux 2.4.x and 2.6.x IPv4 packet filtering ruleset[52]. The *iptables* package also includes *ip6tables* that is used for configuring the IPv6 packet filter.

There are four built-in tables in *iptables*, such as *filter*, *nat*, *mangle* and *raw*. In our experiment, only the first two tables are used. The *filter* is the default table. The filter table should be used exclusively for filtering packets. And the *nat* table should be mainly used for Network Address Translation. Packets in a stream only traverse this table once. If the first packet of a stream is allowed, the rest of the packets in the same stream will be subject to the same actions as the first packet.

As well as user-defined chains, each table contains some predefined chains such as PREROUTING, POSTROUTING, INPUT, FORWARD and OUTPUT. A chain is a series of rules that consists of conditions and an action specifying what to do with a matching packet. An action is called a 'target'. There are a number of actions. For example, ACCEPT target means to let packets traverse through. Besides, a target DROP means to discard packets and will not carry out any further processing. If a packet meets a rule's condition of one chain, the corresponding action specified by the rule's target will be executed. Otherwise, the next rule of the chain will be checked.

As shown in Figure 5-9, the incoming packet flow left to right through one or more chains. Generally, all incoming packets come through the PREROUTING chain of the *nat* table which alters packets as soon as they come in; Incoming packets destined for the local host go through the INPUT chain of the *filter* table, no matter what interface or in which direction they came from; Local outgoing packets traverse through the OUTPUT chain of the *filter* table. Packets to be routed pass through the FORWARD chain of the *filter*; Outgoing packets go through the POSTROUTING

chain of the *nat* table.

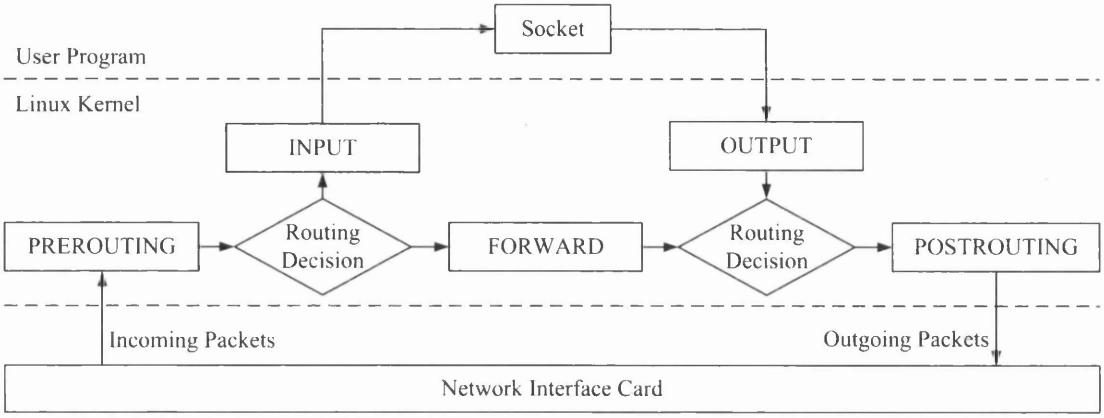

Figure 5-9 Iptables Packet Flow

Therefore, the topology shown in Figure 5-10 can be built by executing the following *iptables* commands at corresponding hosts.

```
Host A:
iptables -t nat -A PREROUTING -s 192.168.182.200 -i eth0 DROP
Host B:
iptables -t nat -A PREROUTING -s 192.168.182.150 -i wlan0 DROP
Host C:
iptables -t nat -A PREROUTING -s 192.168.182.100 -i wlan0 DROP
Host D:
iptables -t nat -A PREROUTING -s 192.168.182.250 -i wlan0 DROP
```

The option *–t* specifies to use the table *nat* rather than the default table (*filter*). Following option *–A* adds one rule to the end of the selected chain. Next specifies a network IP address of the source host of a packet. Optionally, an interface name is explicated. The target indicates what to do with the packet that matches the previous conditions.

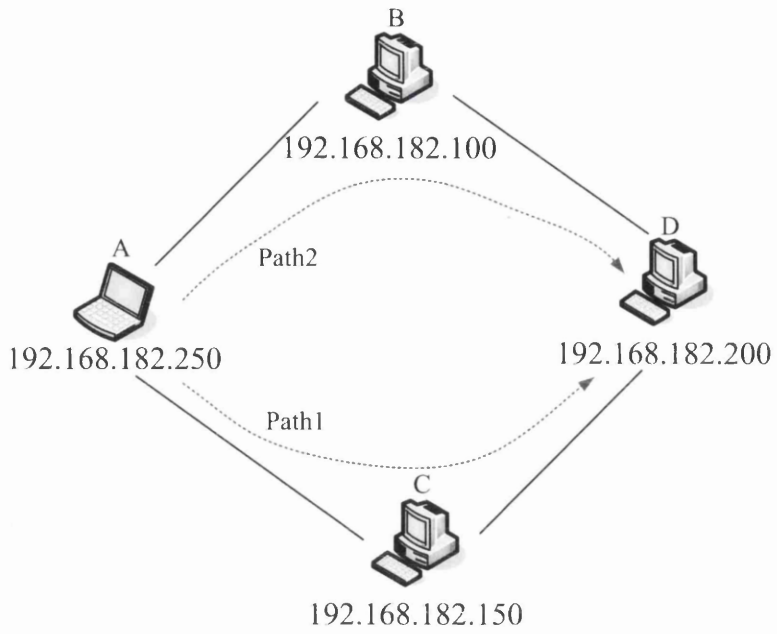

Figure 5-10 Topology of Testbed

5.7.2 Experimental Results

1) Reputation Curve

Assume that host *A* in Figure 5-10 is the source node, and host *D* is the destination. The rest of nodes are responsible for forwarding the control and data packets. The test process is described as following.

At the beginning, after each host connected with its neighbours, the source started to send data packets to the destination by the command *ping*. Without any outside interference, the source randomly selected *C* to forward data to *D*, even though *B* had the same initial global reputation value as *C*. As a result, the reputation of *C* jumped to 1, because so far *C* forwarded all data packets coming from the source host; By contrast, the global reputation of *B* was almost zero as shown in Figure 5-11.

After about one minute, *C* was manually set to deny forwarding data from *A* using *iptables*. Consequently, the global reputation of *B* began dramatically increasing to round 0.45 and the global reputation of *C* decreased to approximately 0.2. After the node *B*'s global reputation surpassed the one of *C*, source node *A* started to choose *B* to forward packets rather than *C*, thus *B*'s global reputation increased further.

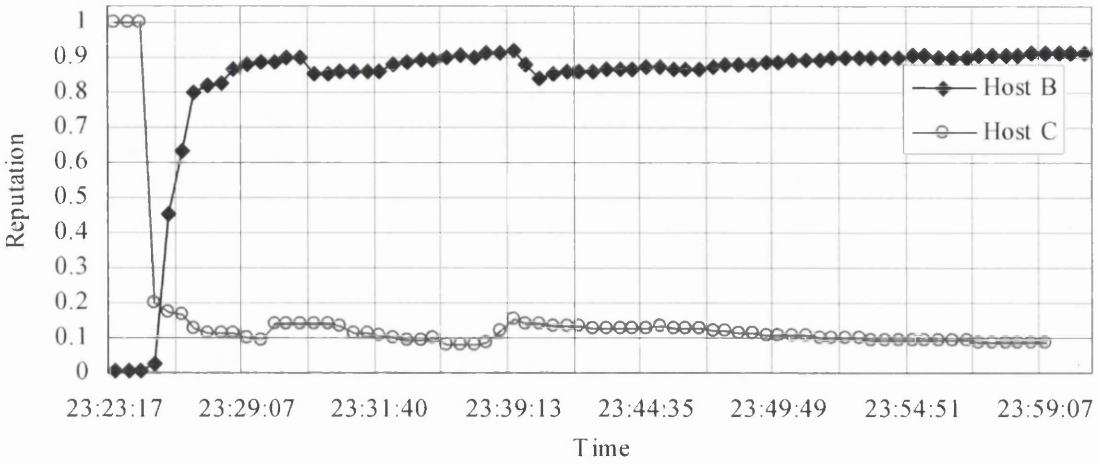

Figure 5-11 Global Reputations of Host B and C

It can be seen that even though at first *A* selects *C* to forward data, but, due to *C* is forced to drop packets from *A* for a long time, *B* whose global reputation begins to increase and finally replaces *C* to forward data to *D*.

Of course, if misbehaviour of *B* causes its own global reputation falling underneath *C*'s global reputation in the future, the route to *C* will be re-selected again. In short, in AODV-ET the source node selects the relative trustful path to forward data packets.

2) Change of Hops

Correspondingly, the change of hops from *A* to *D* in this experiment can be seen from Figure 5-12. At first, the path1 in Figure 5-10 was selected by the source and the distance to *D* was 2 because of no reputation penalty for node *C*. However, misbehavior of *C* resulted in the reduction of its global reputation from 1 to 0.2. According to equation (5-2), the hops of path1 were calculated as follows:

$$
\begin{aligned}
&RH_{AD} = RH_{AC} + RH_{CD} \\
&= [round((1-g_C)\cdot ND)+1]+[round((1-g_D)\cdot ND)+1] \\
&= [round((1-0.2)\times 30)+1]+[round((1-1)\times 30)+1] \\
&= [round(24)+1]+[0+1] \\
&= 26
\end{aligned}
$$

where *ND* measures the maximum possible transmission distance for a routing request in a network, thus its value must be a positive integer. But a large network

diameter can lead to significant delays for route discovery in the network. Here, assume its value is 30 in AODV-ET.

Meanwhile, the reputation of B increased from 0.01 to 0.02, so the hops of path2 were computed as follows:

$$
\begin{aligned}
&RH_{AD} = RH_{AB} + RH_{BD} \\
&= [round((1-g_B)\cdot ND)+1]+[round((1-g_D)\cdot ND)+1] \\
&= [round((1-0.02)\times 30)+1]+[round((1-1)\times 30)+1] \\
&= [round(29.4)+1]+[0+1] \\
&= 31
\end{aligned}
$$

So far, the hops of path1 (26) were less than the one of path2 (31), so the path to the destination still traversed through C even though behavior of C had been abnormal. But when the reputation of B was higher than the one of C, path2 was to be chosen by the source A. Later on, reputation of B jumped from 0.02 to 0.46, thus the hops of path2 fell from 31 to 18, less than the hops of path1 selected before. With the increase of the global reputation of B, hops of the path2 would be further reduced.

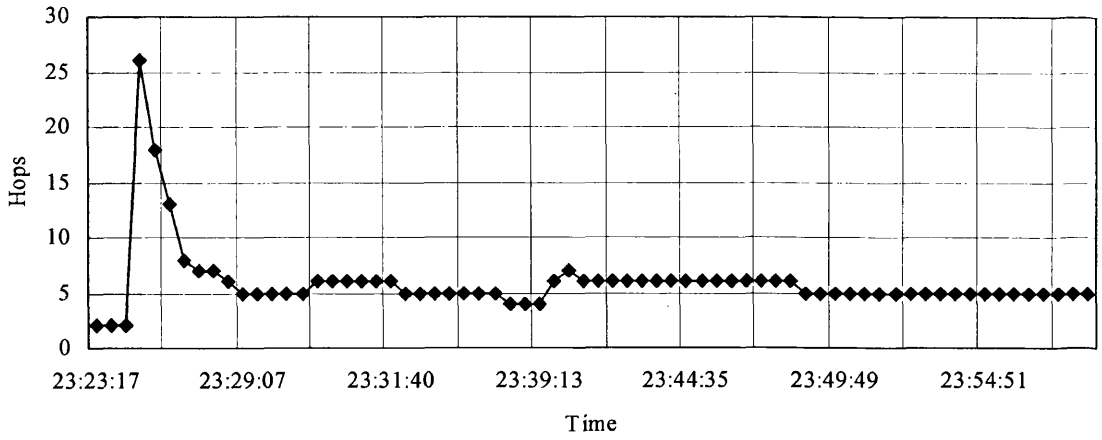

Figure 5-12 Hops from the Source to the Destination

3) Memory Usage

Table 5-3 presents information about memory consumption of two AODV implementations: AODV-UU and AODV-ET. The first column VIRT means the total amount of virtual memory in kilobytes which a task is able to access. It includes all code, data and shared libraries plus memory shared with other tasks. RES stands for the sum size in kilobytes of 'text resident set' and 'data resident set', which is an

accurate representation of how much actual physical memory the task is using. SHR indicates how much of the VIRT size is actually sharable (memory or libraries) by the task.

Table 5-3 Memory Usage Comparison between AODV-UU & AODV-ET

Protocol Implementation	Version	VIRT (KiB)	RES (KiB)	SHR (KiB)
AODV-UU	0.9.6	1756	656	576
AODV-ET	0.1	10396	968	844

According to this Table, AODV-ET consumes nearly 50% more of actual physical memory used by AODV-UU. In addition, the total size of virtual memory consumed by AODV-ET is around 6 times as many as AODV-UU used. Those are because the sniffer module of AODV-ET is based on a few functions in the *Libpcap* library but the library is not used in AODV-UU. As a result, the whole library is mapped and counted in VIRT and SHR of AODV-ET. And the parts of the library file containing the functions being used increase the size of RES of AODV-ET as well.

4) Brief Discussion

In AODV-ET, every node is marked with the global reputation value to represent its reliability in a network. When a node is compromised by the packet drop attack, its global reputation value will be decreased and propagated by the RREQ extension during the routing discovery process. For example, in the above experiment, after node C had been hacked by the attack, its reputation information was transferred to node D with the help of RREQ and stored in D's reputation table. If D attempts to send data to A, it will also know how reliable C is.

Admittedly, AODV-ET increases the routing transmission overhead because of the extension, each of which occupies 128 bits. This could reduce the efficiency of the protocol. On the other hand, compared with the original AODV, AODV-ET protocol has a higher security and could increase the network throughput when a part of nodes

are compromised by the attack.

5.8 Summary

This chapter illustrates the AODV-ET routing protocol and the corresponding experimental procedure. The secure protocol is able to switch from the route compromised by the packet drop attack to the relative trustworthy path for data transmission in our experimental environment. Namely, it can mitigate the effect of the node frequently dropping packets.

6 Conclusions and Future Work

This thesis is concluded herein with a summary of my contributions and directions for future research.

6.1 Conclusions

An EigenTrust based AODV is proposed as an intrusion detection scheme, independently monitoring the routing of data packets and evaluating neighbourhood based on the auditing records, to mitigate the effect of the packet drop attack.

In particular, the EigenTrust algorithm is improved in initial value and termination condition to guarantee its convergence when no pre-trusted nodes are defined in AODV at the cost of reducing the convergence rate. Besides, the application of EigenTrust merges the data fusion layer and the reputation computation layer proposed in [5, 53].

Moreover, the format of RREQ and its corresponding processing procedure are both altered and illustrated in this paper, in order to propagate reputation information and calculate the global reputation.

Furthermore, the reputation based AODV (AODV-ET) is implemented in Linux and tested in an office environment to evaluate its functionality and effectiveness. Experimental results show that AODV-ET is able to switch from the route compromised by the packet drop attack to the relative reliable route.

6.2 Future Work

The feasibility of AODV-ET has been demonstrated in this thesis, but the performance parameters have not been measured, such as network throughput, end-to-end delay and out-of-order delivery percentage. In fact, AODV was designed for large scale ad-hoc networks, so valuable performance parameters of AODV-ET should be accessed by network simulation tools.

According to the Intrusion Detection Message Exchange Format (IDMEF) Communication Protocol Requirements[54], a message's integrity, confidentiality and authenticity should be guaranteed by a proper encryption algorithms. Fortunately, another secure AODV protocol[55], using the RSA algorithm to protect the security of messages in these aspects, has also been developed on the basis of AODV-UU. So, its encryption code could be ported into the AODV-ET protocol for achieving those three security objectives. But, due to time limitation, the work has not been accomplished so far.

Bibliography

[1] Iqbal M, Wang X, Li S, Ellis T. QoS scheme for multimedia multicast communications over wireless mesh networks. *IET Communications* 2010, 4(11):1312-1324.

[2] Wang XH, Iqbal M, Zhou X. Design and implementation of a dual-radio wireless mesh network testbed for healthcare. *5th International Conference on Information Technology and Applications in Biomedicine (ITAB 2008)*. Shenzhen, China, 2008;300-304.

[3] Akyildiz IF, Wang X, Wang W. Wireless Mesh Networks: A Survey. *Computer Networks* 2005, 47(4):445-487.

[4] Kamvar SD, Schlosser MT, Garcia-Molina H. The Eigentrust algorithm for reputation management in P2P networks. *Proceedings of the 12th international conference on World Wide Web*. Budapest, Hungary: ACM, 2003;640-651.

[5] Oliviero F, Peluso L, Romano SP. REFACING: An autonomic approach to network security based on multidimensional trustworthiness. *Computer Networks* 2008, 52(14):2745-2763.

[6] Perkins CE, Royer EM. Ad-hoc on-demand distance vector routing. *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications 1999 (WMCSA '99)* New Orleans, Louisiana, USA: IEEE Computer Society, 1999;90-100.

[7] Jubin J, Tornow JD. The DARPA packet radio network protocols. *Proceedings of the IEEE* 1987, 75(1):21-32.

[8] BelAir. Capacity of Wireless Mesh Networks. Available at: http://www.belairnetworks.com/resources/pdfs/Mesh_Capacity_BDMC00040-C02.pdf. (Accessed 2 Nov 2011 2011)

[9] Lupu T, Rudas I, Demiralp M, Mastorakis N. Main types of attacks in wireless sensor networks. *Proceedings of the 9th WSEAS international conference on signal, speech and image processing*. Budapest, Hungary: WSEAS, 2009;180-185.

[10] Scarfone K, Dicoi D, Sexton M, Tibbs C. Guide to Securing Legacy IEEE 802.11 Wireless Networks. *NIST Special Publication* 2008, 800:48.

[11] Raza MH, Hughes L, Raza I. Density: A Context Parameter of Ad Hoc Networks. *Trends in Intelligent Systems and Computer Engineering* 2008, 6:525-540.

[12] Jing D, Han R, Mishra S. Countermeasures Against Traffic Analysis Attacks in Wireless Sensor Networks. *The 1st International Conference on Security and Privacy for Emerging Areas in Communications Networks*. Athens, Greece: SecureComm, 2005;113-126.

[13] Maheshwari R, Jie G, Das SR. Detecting Wormhole Attacks in Wireless Networks Using Connectivity Information. *26th IEEE International Conference on Computer Communications*. Anchorage, Alaska, USA: IEEE, 2007;107-115.

[14] GCHQ. Public-Key Encryption. Available at: http://www.gchq.gov.uk/history/pke.html. (Accessed 11 Nov 2011)

[15] R. H. *The architecture of a network-level intrusion detection system*: Dept. of Computer Science, College of Engineering, University of New Mexico; 1990.

[16] Chen T, Kuo GS, Li ZP, Zhu GM. Intrusion detection in wireless mesh networks. *Security in wireless mesh networks* 2008):145.

[17] Poblete O. An Overview of the Wireless Intrusion Detection System. Available at:

http://www.sans.org/reading_room/whitepapers/wireless/overview-wireless-intrusion-detection-system_1599. (Accessed 3 Nov 2011)

[18] Lemonnier-Defcom E. Protocol Anomaly Detection in Network-based IDSs. Available at: http://www.sans.org/reading_room/whitepapers/detection/protocol-anomaly-detection-network-based-intrusion-detection_349. (Accessed 3 Nov 2011)

[19] Ho SY. Intrusion Detection-Systems for today and tomorrow. Available at: http://www.sans.org/reading_room/whitepapers/detection/intrusion-detection-systems-today-tomorrow_341.

[20] Zhang Y, Lee W, Huang YA. Intrusion detection techniques for mobile wireless networks. *Wireless Networks* 2003, 9(5):545-556.

[21] Johnson DB, Maltz DA, Broch J. The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR). Available at: http://www.cs.cmu.edu/~dmaltz/internet-drafts/draft-ietf-manet-dsr-09.txt. (Accessed 15 Dec 2011)

[22] Marti S, Giuli TJ, Lai K, Baker M. Mitigating routing misbehavior in mobile ad hoc networks. *Proceedings of the 6th annual international conference on Mobile computing and networking*. Boston, Massachusetts, United States: ACM, 2000;255-265.

[23] Trivedi AK, Kapoor R, Arora R, Sanyal S. Rism-reputation based intrusion detection system for mobile adhoc networks. *3rd International Conference on Computers and Devices for Communication (CODEC-06)* 2006. Vol. 6.

[24] Fenye B, Ing-Ray C, MoonJeong C, Jin-Hee C. Trust-Based Intrusion Detection in Wireless Sensor Networks. *IEEE International Conference onCommunications (ICC)*. Kyoto, Japan: IEEE, 2011;1-6.

[25] Asad Amir P, Chris M. Establishing trust in pure ad-hoc networks. *Proceedings of the 27th Australasian conference on Computer science*. Dunedin, New Zealand: Australian Computer Society, Inc., 2004.

[26] Buchegger S, Boudec J-YL. Performance analysis of the CONFIDANT protocol. *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking*. Lausanne, Switzerland: ACM, 2002;226-236.

[27] Rafsanjani MK, Movaghar A, Koroupi F. Investigating Intrusion Detection Systems in MANET and Comparing IDSs for Detecting Misbehaving Nodes. *World Academy of Science, Engineering and Technology* 2008):44.

[28] Michiardi P, Molva R. Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. *Proceedings of the IFIP TC6/TC11 6th Joint Working Conference on Communications and Multimedia Security: Advanced Communications and Multimedia Security*. Portoroz, Slovenia: Kluwer B.V., 2002;107-121.

[29] Qi H, Dapeng W, Khosla P. SORI: a secure and objective reputation-based incentive scheme for ad-hoc networks. *IEEE Wireless Communications and Networking Conference (WCNC 2004)*. Atlanta, GA, USA, 2004;825-830.

[30] Shafer G. *A mathematical theory of evidence*. Vol. 1: Princeton University Press; 1976.

[31] Wang P, Yang G. Improvement method for the combining rule of Dempster-Shafer evidence theory based on reliability. *Journal of Systems Engineering and Electronics* 2005, 16(2):471-474.

[32] Oliviero F, Romano SP. A Reputation-Based Metric for Secure Routing in Wireless Mesh Networks. *IEEE Global Telecommunications Conference 2008 (GLOBECOM 2008)*. New Orleans, Louisiana, USA, 2008;1-5.

[33] Guillaume L, van de Sype J, Schumacher L, Di Stasi G, Canonico R. Adding reputation

extensions to AODV-UU. *2010 17th IEEE Symposium on Communications and Vehicular Technology in the Benelux (SCVT 2010)*. Netherlands, 2010;1-6.

[34] Kumar S, Parthipan V. SOPE: Self-organized protocol for evaluating trust in MANET using Eigen Trust Algorithm. *The 3rd International Conference on Electronics Computer Technology (ICECT)*. Kanyakumari, India: IEEE, 2011;155-159.

[35] Haveliwala T, Kamvar S. The Second Eigenvalue of the Google Matrix. Available at: http://ilpubs.stanford.edu:8090/582/. (Accessed 23 Nov 2011)

[36] Newsham T, Ptacek T. Insertion, Evasion and Denial of Service: Eluding Network Intrusion Detection. *Secure Networks, Inc. Technical Report, January* 1998.

[37] Chen TM, Venkataramanan V. Dempster-Shafer theory for intrusion detection in ad hoc networks. *Internet Computing, IEEE* 2005, 9(6):35-41.

[38] Wei H, Jianhua L, Qiang G. Intrusion Detection Engine Based on Dempster-Shafer's Theory of Evidence. *2006 International Conference on Communications, Circuits and Systems Proceedings*. Chengdu, China: IEEE, 2006;1627-1631.

[39] Wang P. The Reliable Combination Rule of Evidence in Dempster-Shafer Theory. *2008 International Congress on Image and Signal Processing*. SanYa, Hainan, China: IEEE Computer Society, 2008;166-170.

[40] Zadeh LA. Review of A Mathematical Theory of Evidence. *AI magazine* 1984, 5(3):81.

[41] Golub GH, Van der Vorst HA. Eigenvalue computation in the 20th century. *Journal of Computational and Applied Mathematics* 2000, 123(1-2):35-65.

[42] Mises RV, Pollaczek-Geiringer H. Praktische Verfahren der Gleichungsauflösung. *ZAMM - Zeitschrift für Angewandte Mathematik und Mechanik* 1929, 9(1):58-77.

[43] Jantsch E. *The self-organizing universe: Scientific and human implications of the emerging paradigm of evolution*: Pergamon Press Oxford; 1980.

[44] Brookes M. The Matrix Reference Manual. Available at: http://www.ee.ic.ac.uk/hp/staff/dmb/matrix/special.html#Reducible. (Accessed 12 Dec 2011)

[45] Gihan Nagib, Ali WG. Network Routing Protocol using Genetic Algorithms. *International Journal of Electrical & Computer Sciences* 2010, 10(02):4.

[46] Tim Carstens, Harris G. Programming with pcap. Available at: http://www.tcpdump.org/pcap.html. (Accessed 16 Nov 2011)

[47] Steven M, Van J. The BSD packet filter: a new architecture for user-level packet capture. *Proceedings of the USENIX Winter 1993 Conference Proceedings on USENIX*. San Diego, California: USENIX Association, 1993;2.

[48] Nordstrom E, Lundgren H. AODV-UU Implementation from Uppsala University. Available at: http://aodvuu.sourceforge.net/. (Accessed 23 Sep 2011)

[49] Wiberg B. Porting AODV-UU implementation to ns-2 and Enabling Trace-based Simulation. *UPPSALA University Master Thesis in Computer Science* 2002.

[50] Nospam R. Installation System Requirements. Available at: https://help.ubuntu.com/community/Installation/SystemRequirements. (Accessed 28 Oct 2011)

[51] Team G. The GNU Compiler Collection. Available at: http://gcc.gnu.org/. (Accessed 30 Dec 2011)

[52] Ikeda M, Hiyama M, Barolli L, Xhafa F, Durresi A, Takizawa M. Mobility Effects of Wireless Multi-hop Networks in Indoor Scenarios. *2010 24th IEEE International*

Conference on Advanced Information Networking and Applications (AINA). Perth, Australia: CPS495-502.

[53] Buchegger S, Mundinger J, Le Boudec JY. Reputation Systems for Self-Organized Networks. *Technology and Society Magazine, IEEE* 2008, 27(1):41-47.

[54] Wood M. Intrusion Detection Message Exchange Requirements. Available at: http://tools.ietf.org/html/rfc4766#page-12. (Accessed 24 Dec 2011)

[55] Cerri D, Ghioni A. Securing AODV: the A-SAODV Secure Routing Prototype. *Communications Magazine, IEEE* 2008, 46(2):120-125.