ABSTRACT

Title of Document: TRUST-BASED DEFENSE AGAINST INSIDER PACKET DROP ATTACKS IN WIRELESS SENSOR NETWORKS

Youngho Cho, Doctor of Philosophy, 2013

Directed By: Professor Gang Qu
Department of Electrical and Computer Engineering

In most wireless sensor networks (WSNs), sensor nodes generate data packets and send them to the base station (BS) by multi-hop routing paths because of their limited energy and transmission range. The insider packet drop attacks refer to a set of attacks where compromised nodes intentionally drop packets. It is challenging to accurately detect such attacks because packets may also be dropped due to collision, congestion, or other network problems.

Trust mechanism is a promising approach to identify inside packet drop attackers. In such an approach, each node will monitor its neighbor's packet forwarding behavior and use this observation to measure the trustworthiness of its neighbors. Once a neighbor's trust value falls below a threshold, it will be considered as an attacker by the monitoring node and excluded from the routing paths so further damage to the network will not be made.

In this dissertation, we analyze the limitation of the state-of-the-art trust mechanisms and propose several enhancement techniques to better defend against insider packet drop attacks in WSNs.

First, we observe that inside attackers can easily defeat the current trust mechanisms and even if they are caught, normally a lot of damage has already been made to the network. We believe this is caused by current trust models' inefficiency in distinguishing attacking behaviors and normal network transmission failures. We demonstrate that the phenomenon of consecutive packet drops is one fundamental difference between attackers and good sensor nodes and build a hybrid trust model based on it to improve the detection speed and accuracy of current trust models.

Second, trust mechanisms give false alarms when they mis-categorize good nodes as attackers. Aggressive mechanisms like our hybrid approach designed to catch attackers as early as possible normally have high false alarm rate. Removing these nodes from routing paths may significantly reduce the performance of the network. We propose a novel false alarm detection and recovery mechanism that can recover the falsely detected good nodes.

Next, we show that more intelligent packet drop attackers can launch advanced attacks without being detected by introducing a selective forwarding-based denial-of-service attack that drops only packets from specific victim nodes. We develop effective detection and prevention methods against such attack.

We have implemented all the methods we have proposed and conducted extensive simulations with the OPNET network simulator to validate their effectiveness.

TRUST-BASED DEFENSE AGAINST INSIDE PACKET DROP ATTACKS IN WIRELESS SENSOR NETWORKS

By

Youngho Cho

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2013

Advisory Committee:
Professor Gang Qu, Chair/Advisor
Professor Richard La
Professor K. J. Ray Liu
Professor Charles Silio
Professor Lawrence C. Washington

Dedication

To my family

Acknowledgements

First, I would like to thank my advisor Professor Gang Qu for having guided me with his kindness and wisdom in the last five years. Without his encouragement and support, I would not be able to overcome the challenges and difficulties that I have faced during my Ph.D. study in the US. It is my best luck to meet such a great and trustful advisor in the US. I deeply appreciate his support and belief on me.

I also would like to thank Professor Richard La, Professor K. J. Ray Liu, Professor Charles Silio, and Professor Lawrence C. Washington for kindly agreeing to serve on my dissertation committee, spending their time on reviewing my dissertation manuscript, and providing valuable comments and suggestions.

I am grateful for the Republic of Korea Air Force (ROKAF) for its support on my Ph.D. study at the University of Maryland, College Park, USA. I have also been supported by the Air Force Office of Scientific Research (AFOSR/RSL) under Award No. #FA95501010140 and a University Partnership with the Laboratory of Telecommunications Sciences, Contract Number H9823013D00560002.

I have to thank many friends and colleagues who share joyful and hard time with me during my stay in the US: Dr. Kisung Um and Yoonjung Yang, Ikhwan Do, Dr. Donghyun Cho, Sung Jun Yun, Young Wook Kim, Hyun Soo Kim, Taek Il Oh, Sangjin Han, Doo-Hyun Sung, Jonghyun Choi, Ginnah Lee, Shinkyu Park, Scott Kim, Tim Creech, Dr. Jounghoon Bae, Dr. Soo Bum Lee, Dr. Dongwoon Hahn, Dr. Sangchul Song, Dr. Minkyung Cho, Dr. Hsiang-Huang Wu, Susan Gould, Marci and Will Martin, Katie Cote, Ji-Eum and Ha-Sim cell members at GMC. Especially, Dr. Kye-Hwan Kim (my senior officer at the ROKAF) with whom I went through every step together in this

long journey at the UMCP. We shared many unforgettable moments in the US. I believe we will be not only best colleagues but also best friends for the rest of our remaining journey. Hyungtae Lee joined the same Ph.D. program in 2008, sharing both the most painful and joyful time with me. I will remember the time we went to Annapolis after I received the email notifying I finally passed the Ph.D. qualifying exam. Zheng Zhou and I discussed many of my research ideas while having coffee in the A.V.W. building. He is always enthusiastic in learning my research ideas and gives me many useful comments. I really enjoyed talking with him about our family and future. Norma and Byron are invaluable persons for our family. They took care of my family as parents, especially when I was too busy to take care of my family. I saw God's love through them. I would like to thank many reviewers and editors, especially John Kuntz. I have to thank many other friends and colleagues for their love and encouragement although I could not enumerate each of their names here.

I would like to thank to OPNET Technologies, Inc, for providing me with the OPNET Wireless Modeler to validate my proposed approaches.

Most of all, I give my best appreciation to my family for their pure love for me: My wife Joungyun Ji for her sacrifice, support and love, my adorable son Ryan Junyoung Cho for recharging my battery with his mesmerizing smile, my sister Jeonghye and her family and my parents for their warm love, my parents-in-law for their sincere prayers, and all my relatives for wishing me luck and happiness.

Finally, I thank God for everything He has prepared for me.

A.V.W. Building, College Park, MD, USA

Youngho Cho, July 2013

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ACT	Avoidance Completion Time
ACK	Acknowledgement
BS	Base Station
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
DCT	Detection Completion Time
DoS	Denial of Service
DR	Detection Rate
FADER	False Alarm Detection and Recovery
FAR	False Alarm Rate
FCFS	First Come First Serve
FS	Forwarding Set
FSM	Finite State Machine
GPSR	Greedy Perimeter Stateless Routing
GRP	Geographic Routing Protocol
IDS	Intrusion Detection System
MEMS	Micro-Electro-Mechanical Systems
NS	Neighbor Set
NSA	Neighbor Selection Algorithm
PDR	Packet Delivery Rate
PKI	Public Key Infrastructure
SNMT	Source Neighbor Mapping Table
WSN	Wireless Sensor Network

Chapter 1

Introduction

1.1 Insider Packet Drop Attacks in Wireless Sensor Networks

With the rapid advancement of semiconductor and MEMS (Micro-Electro-Mechanical Systems) technologies, miniature wireless sensor nodes are becoming more powerful and less expensive with better mobility. Wireless sensor networks (WSNs) are widely used in many practical applications including military applications, disaster detection and prevention, forest fire protection, habitat monitoring, smart space, etc [19, 24, 56]. In general, hundreds and thousands of such small inexpensive wireless sensors (with a limited transceiver range and battery) are deployed in the field to collect and report data to the destination such as the base station (BS) or the fusion center in a collaborative multi-hop fashion (*multi-hop communication*). Here, it is critical that each intermediate node (or relay node) will collaborate by forwarding the received packets to the next hop neighbor toward the destination. However, since the infrastructure of the WSN (both the sensor nodes and their connectivity) may not be trusted, can create security concerns and cause damage to the performance of the network.

Insider threat is one of the most important and challenging security issues in WSNs because of its following unique characteristics [7, 15, 27, 57]. First, many WSNs are deployed and operated in harsh environments without human involvement. Thus, it is possible that adversaries can physically access the wireless sensors and maliciously manipulate or compromise them to inside attackers. Second, these inside attackers are legitimate members of the network and will not be caught by authentication and

authorization mechanisms that are very effective in preventing outside attackers from launching attacks such as eavesdropping or packet modifications. Third, due to the multi-hop packet routing nature of the WSNs, an inside attacker may receive data packets from many other sensors, it can disrupt network operations significantly by modifying packet information or simply dropping critical packets. Fourth, a wireless sensor has limited resources such as memory size, battery power, computing capability, transmission range, etc. Therefore, the legacy but expensive security mechanisms such as PKI (Public Key Infrastructure) and heavy IDS (Intrusion Detection System) used in traditional wired or wireless networks are not suitable for many WSNs. This restriction makes it more difficult to defend against inside attackers. Finally, an inside attacker, as a member of the WSN, know the security mechanisms being used in the WSN and can learn the vulnerabilities of these security mechanisms. It can exploit the discovered vulnerabilities to break the security mechanisms or launch sophisticated attacks without being caught.

Inside attackers can launch various types of attacks actively (such as modification, packet drop, or misrouting) or passively (such as eavesdropping). Among these, insider packet drop attacks is arguably the easiest to launch, most difficult to detect, yet could cause significant network performance degradation, especially when the inside packet drop attackers are well-positioned in the network (e.g., close to the BS or at the intersection of many routing paths) [2, 7, 10]. Inside packet drop attackers drop packets to cause a serious damage to the network either by dropping as many packets as possible to significantly degrade network performance or by preventing critical packets from being delivered to their destination (Denial-of-Service attack or DoS attack in short). Moreover, it is very tricky to defend against insider packet drop attacks because, for a

particular packet drop, we cannot distinguish whether it is dropped by an attacker or as a result of a natural network problem such as collision, interference, noise, fading, obstacles, etc [10, 32]. There are several types of packet drop attacks such as blackhole attacks, grayhole attacks (a.k.a., selective forwarding attacks), and on-off attacks [1, 7, 15], which we will elaborate in Chapter 2. Blackhole attackers drop all the received packets, on-off attackers drop packets following a rather fixed on and off pattern, and grayhole attackers have more complicated and irregular attacking patterns and are the more challenging attackers to defend. For this reason, insider packet drop attacks, particularly grayhole attacks, have attracted a lot of attention and will be the focus of this dissertation.

Current defending approaches against insider packet drop attacks are either *avoidance approach* or *detection approach*. The avoidance approaches focus on how to deliver the packets successfully with the existence of the attackers. They are not designed to catch the attackers. A popular way to achieve this is to use multiple disjoint routing paths [7, 48, 58, 59]. In [7], the authors pointed out that k disjoint multipath routing can completely defend against selective forwarding attacks when there are no more than $k-1$ compromised nodes. Similarly, a multiple data flow scheme using multiple disjoint topologies was introduced in [5]. In this scheme, the source node sends its packet through one or more topologies randomly chosen from the pre-established multiple topologies to avoid packet drop attackers. However, all the avoidance approaches have very high cost in terms of network traffic, transmission energy, etc [4].

As a representative detection approach, trust mechanism with the notion of trust in human society has been proven as a promising approach to identify inside packet drop

attackers [1, 6, 12, 15, 22, 25]. In this approach, each sensor node will monitor its neighbor's packet forwarding behavior and use this observation to measure the trustworthiness of its neighbors. Once a neighbor's trust value falls below a pre-determined threshold, the monitoring node will consider this neighbor as an inside attacker and exclude it from the routing paths to avoid further damage to the network. In general, the trust mechanism works in the following three stages: 1) node behavior monitoring, 2) trust measurement, and 3) insider attack detection. Watchdog [10] that uses a sensor's overhearing ability is a widely used direct monitoring mechanism used in the first stage. The other two stages are processed by a trust model such as Beta trust model [6] and Entropy trust model [12] using the data collected by the watchdog. A trust-based routing that combines a trust mechanism with a routing algorithm is used for packet routing in a WSN. Many researchers [12, 16, 22, 25, 71, 72, 73] have showed that trust-based routing approaches can gracefully mitigate insider packet drop attacks by building trusted routing paths to the destination. Moreover, they showed that trust-based routing improves packet's successful delivery rate under insider attacks over routing algorithms that do not consider trust.

Clearly, the effectiveness of these trust-based routing protocols is based on the underlying trust model. A good trust model will help the routing algorithm to quickly and accurately identify inside attackers and find alternate routes to avoid them. Moreover, intelligent inside attackers can adjust their attacking strategy based on the detection mechanisms used in the WSN to launch new and more powerful attacks. In this dissertation, we will analyze the limitations of the state-of-the-art trust mechanisms and

propose several enhancement techniques to better defend against insider packet drop attacks in WSNs.

In the remaining of this introduction chapter, we will highlight the three problems we have studied and the main contributions we have made as illustrated in Figure 1.1. Detailed discussions are given in Chapters 3, 4, and 5, respectively. Chapter 2 provides the background of the work and surveys current approaches. Chapter 6 summarizes the dissertation work and Chapter 7 outlines several future research directions.

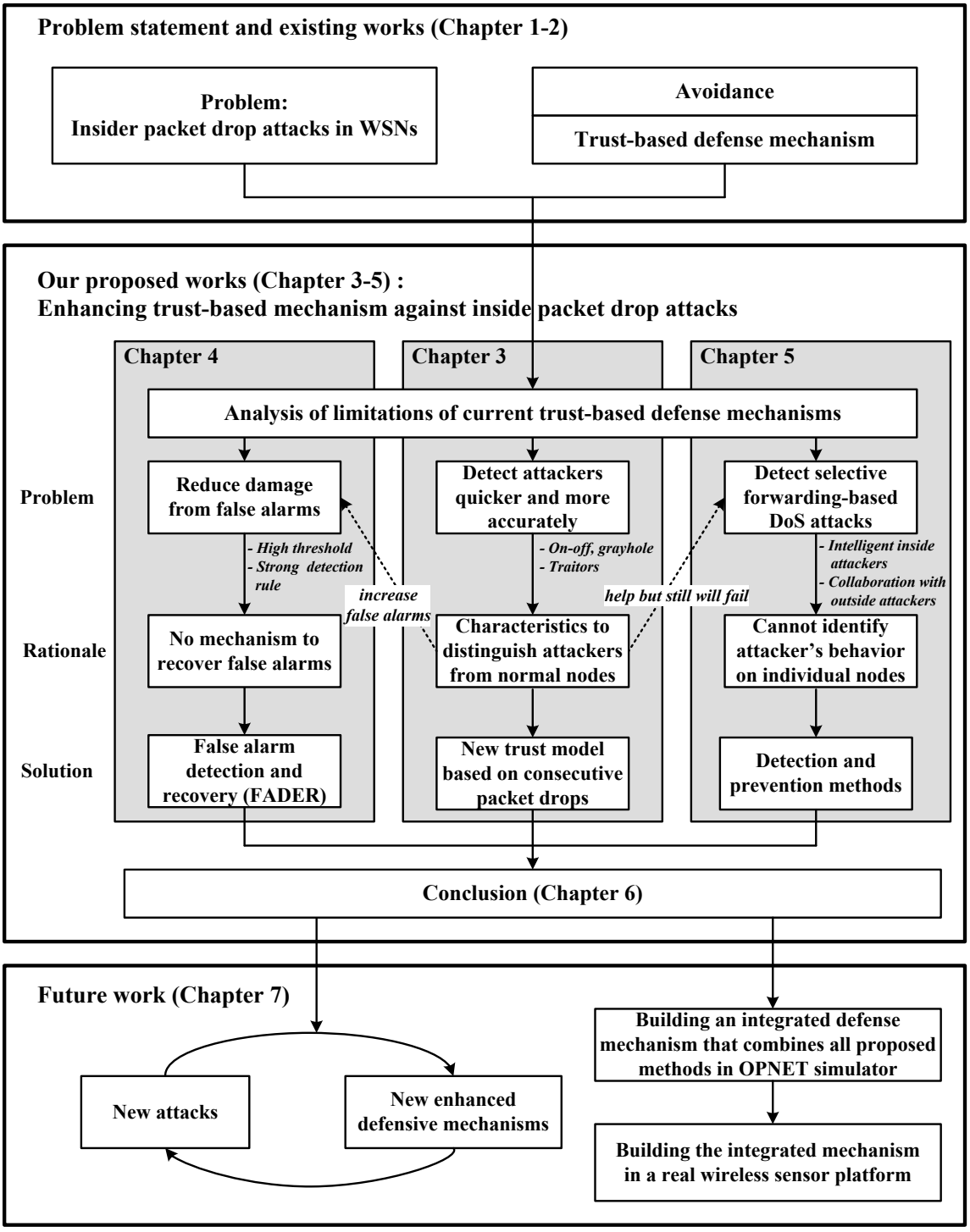


Figure 1.1: The big picture of this dissertation.

1.2 Statement of Problems

In this dissertation, we study the following three critical problems in the domain of trust-based defense against insider packet drop attacks in WSNs.

1) Fast and accurate detection of inside packet drop attackers

In the presence of inside packet drop attackers in the network, to minimize the damage caused by the attackers, we must detect them as fast as possible once they start launching the attacks. In the trust mechanism, a monitoring sensor node will lower the trust value of the node it monitors once it observes packet drops and then consider the node as an attacker if the trust value becomes lower than a threshold. Meanwhile, a normal node (non-attacker) may be observed dropping packets due to network problems such as collision, interference, fading, etc. Therefore, it is critical to design a trust model with a high attacker detection rate and a low false alarm rate. However, this is not a trivial problem because it is hard to distinguish whether a packet drop is caused by attacks or by other natural reasons in WSNs. Moreover, a detection method should not be computational expensive because it is deployed in a wireless sensor with limited resources. This is the reason why current trust models (such as the beta trust model and the entropy trust model) used in WSNs are all designed in a simple way to be feasible for use on a wireless sensor. For example, the beta trust model needs only two counters and a couple of simple operations (addition and division) to measure the trust value of a node (see equation (2.5)), as we will discuss in Chapter 2. However, due to this requirement of simplicity, current trust models are unable to quickly detect inside packet drop attackers, particularly when they have previously earned high trust value (called *traitors*). Therefore, ***we propose to study how to design a more efficient trust model to improve the attack***

detection speed of the current lightweight trust models without significantly increasing overhead and false alarms.

2) Mitigation of false alarms in trust-based routing

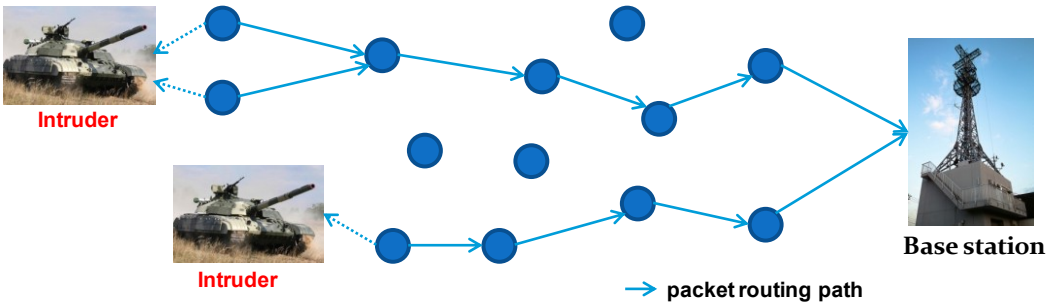
The effectiveness of trust-based routing protocols is based on their underlying trust models. A good trust model will help the routing algorithm to quickly and accurately identify inside packet drop attackers and find alternate routes to avoid them. However, as we have mentioned, it is challenging to distinguish the natural packet drops due to network problems from the intentional packet drops by the inside attackers. No trust model can detect all the inside attackers (100% detection rate) and not misclassify any of the good nodes (0% false alarm rate) [28]. While most of the existing approaches focus on how to improve the detection rate without paying much attention on the false alarm rate, ***we propose to study what to do when false alarm happens, specifically when the false alarm rate is high.*** The rationale behind this proposed study is as follows: First, natural packet drops do occur in a WSN due to fading, interference, collision, noise, congestion, and others [10, 32]. In certain circumstances, such as weather change or moving objects, there may be an abnormality in the WSN's operating environment, which will cause non-negligible amount of packet drops and result in false alarms. Second, once a node is considered as untrustworthy or as an attacker, the existing trust-based routing mechanisms will not use this node any more. In the case of a false alarm, this will affect the routing performance. It also shortens the network lifetime as it is well-known that network's lifetime is closely correlated to the time when the first node in the network goes down [31, 39, 40]. Therefore, the proposed study on how to mitigate false

alarms will be complementary to and have significant impact on the performance of any trust-based routing algorithm.

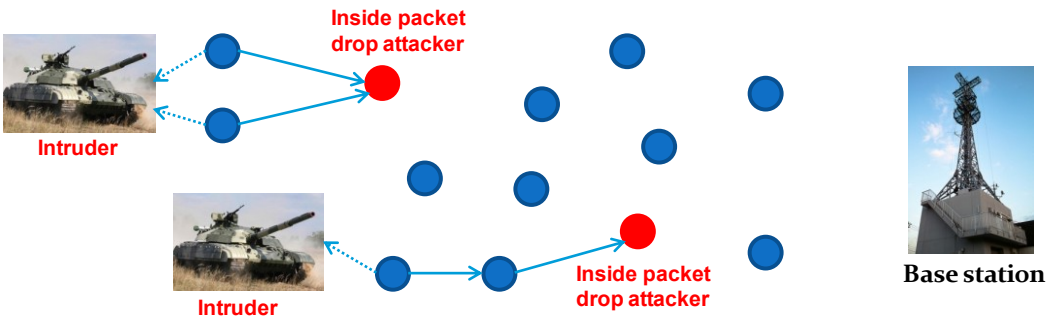
3) Defense against intelligent insider packet drop attacks

We consider that an intelligent packet drop attacker can estimate its trust value and exploit its internal knowledge about the network and the trust mechanism including the trust threshold. In addition, such an attacker can use these estimates to adjust its packet drop rate without being detected by the network's trust mechanism. For example, when its estimates that its trust value is close to the trust threshold, the attacker can temporarily stop dropping the victim node's packets and start forwarding all the packets, including the victim node's, in order to increase its trust value and earn trust from its neighbors and the victim node. Once the attacker earns high trust value at the victim node again, it can repeat the packet drop attacks against the victim node again. In this case, trust mechanism will not be able to detect this intelligent packet drop attacker. We introduce such an intelligent packet drop attack, so called *selective forwarding-based denial-of-service (DoS) attack*, where the attacker drops only some source sensor nodes' data packets of interest in order to prevent their packets from reaching the BS and thus lead to denial of service against the physical area monitored by those sensor nodes. As a motivation for the importance of studying selective forwarding-based DoS attacks, we consider a WSN deployed in a territory for intruder detection. With the help of insiders that perform the selective forwarding-based DoS attack, an outside intruder will be able to enter the territory from the area monitored by victim nodes (to the selective forwarding-based DoS attacks) without being noticed by the BS, as shown in Figure 1.2. ***We propose to study how to***

improve the existing trust mechanisms so that they can effectively defend against the selective forwarding-based DoS attacks.



(a) When there are no inside packet drop attackers



(b) When there are inside packet drop attackers (red nodes)

Figure 1.2: A motivational example of insider packet drop attacks. In (a), outside intruders (tanks) are detected by the base station. In (b), two inside packet drop attackers (red nodes) are dropping critical packets to protect outside intruders.

1.3 Key Contributions

In this dissertation, we study the above important problems in the domain of trust-based defense against insider packet drop attacks and make the following key contributions:

1) We demonstrate that the phenomenon of consecutive packet drops is one fundamental difference between attackers and good sensor nodes and build a hybrid

trust model based on it to improve the detection speed and accuracy of current trust models: In Chapter 3, based on our analysis of goals of insider packet drop attacks and existing packet drop attack models, we propose that the occurrence of consecutive drops is a promising feature for fast and accurate detection of inside packet drop attackers. When an inside packet drop attacker is dropping packets consecutively, we can probabilistically measure the degree of abnormality (or maliciousness) of the node, and penalize the node based on it. However, existing trust models, such as the beta trust model and entropy trust model, do not consider the consecutive drops for trust evaluation. We design a lightweight new trust model based on the consecutive drops, and build a hybrid trust model that can adaptively choose between this new trust model and the beta trust model according to the number of consecutive drops. Extensive simulations based on the widely used commercial network simulator OPNET Wireless Modeler [21] show that our hybrid trust model outperforms the beta trust model in terms of important network performance parameters, such as attack detection speed, attack detection accuracy, routing reliability, and energy-efficiency. Specifically, the results show that our hybrid trust model can always detect various inside packet drop attackers faster than the beta trust model. In addition, due to the fast attack detection capability based the consecutive drops, a geographic routing protocol with our hybrid trust model (Hybrid GRP) can deliver approximately 92% of data packets from source nodes to the BS successfully even when 15% of nodes in the network are grayhole attackers that randomly drop 30% of packets while the geographic routing (Pure GRP) and the geographic routing protocol with the beta trust model (Beta GRP) can deliver only 71% and 78% of packets to the BS, respectively. Moreover, our Hybrid GRP consumes less

energy to send a packet to the BS than the Pure GRP and the Beta GRP.

2) We devise a false alarm detection and recovery mechanism (FADER) that mitigates false alarms by re-evaluating the trustworthiness of the falsely detected good nodes and re-determining whether they are attackers: As described in Chapter 4, our main idea to treat false alarms in a trust-based routing is to give these nodes a second chance such that their neighbors can continue to monitor their behavior and evaluate their trustworthiness. If their neighbors are convinced that false alarm has occurred, these nodes will be re-considered as good nodes and play their normal role in routing. However, if these nodes are confirmed to be inside attackers, they will be eliminated from the routing table just like the current trust-based routing algorithms. We call this mechanism False Alarm DEtection and Recovery, or FADER in short. The key contribution in FADER is that it allows a node to detect and correct (by recovery) its false alarm. By doing so, we can improve the performance of the trust-based routing protocol in terms of many metrics, such as the network lifetime, the packet delivery rate, and many routing performance measures. We have conducted extensive OPNET simulations and the results confirm these claimed advantages of our proposed FADER approach over a representative trust-based routing algorithm. The results show that FADER is able to recover 60−70% of the false alarms without recovering any of the attackers in the presence of multiple attackers in lossy WSNs. In addition, with the false alarm detection and recovery feature, FADER can improve the network lifetime by 18.7−82.9% of Beta GRP and reduce the average hops per path by 7.5−16.7% of Beta GRP.

3) We demonstrate how intelligent inside packet drop attackers can successfully launch denial-of-service attacks against many sensor nodes without being detected by

current trust models (called the selective forwarding-based DoS attack) and propose effective detection and prevention methods against such attack: In Chapter 5, we first describe a simple selective forwarding-based DoS attack and show that the popular trust-based approaches (such as the beta trust model and the entropy trust model) for inside attacker detection fail to detect such an attack. We also analyze the potential damage this attack can cause to the network. We then propose a source-level trust evaluation scheme to enhance the beta and entropy trust mechanisms for effective detection of the selective forwarding-based DoS attackers. Once the attacker is identified, we propose two avoidance strategies to re-route the victim's packets so they can reach the BS. We conduct extensive OPNET simulations to validate our claims and demonstrate the advantages of our proposed approaches. The results show that the existing trust models (the beta trust model and the entropy trust model) with our detection and avoidance approaches with small energy and memory overhead can detect selective forwarding DoS attackers and avoid them when the same trust models without our methods failed to detect the attackers. In addition, as a complementary defensive mechanism to our detection and avoidance methods, we also introduce a prevention routing algorithm to proactively prevent the selective forwarding-based DoS attacks. The key idea of our prevention method is to limit the number of source nodes from which a node receives packets through the same monitoring node. As a result, our prevention method forces the attacker to make a choice: attack and being caught or not attack. The OPNET simulation results show that the beta trust model with our prevention method successfully defend against a selective forwarding-based DoS attacker while the beta trust model fails in detecting the attacker.

Chapter 2

Background and Related Work

2.1 Packet Forwarding Procedure in WSNs

To understand the behaviors of inside packet drop attackers, we briefly explain how a wireless sensor node forwards its data packet to another node toward the BS. We consider the scenario that a node A delivers its own packet (A is a source node) or a received packet from other nodes (A is a relay node) to the base station (BS) in a WSN. The packet forwarding procedure in a WSN follows CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance), which is a wireless channel access method used in wireless networks where A node can transmit its packet only when the channel is idle [41, 52, 54]. That is, if another node is transmitting on the channel within its wireless range (channel is busy), the node needs to wait for a random period of time (called *random back off time*).

When node A has a packet to forward toward the BS, which is stored in its packet forwarding queue $\boldsymbol{Q_F}$, A forwards the packet toward the BS according to the packet forwarding procedure as shown in Figure 2.1. The routing path from A to BS is constructed following some routing algorithm that allows multi-hop communication for energy efficiency. For each intermediate node i in the routing path, let $\mathbf{NS_i}$ be its Neighbor Set consisting of all its neighbor nodes and $\mathbf{FS_i}$ (Forwarding Set) be the set of candidate nodes which the packet may be forwarded to. Apparently, **FS** is a subset of **NS**. To forward packets to BS, node A first chooses a node B from its $\mathbf{FS_A}$ as the next hop and sends the packet to B. Once the next hop node is determined, A forwards its packet to B

when the channel is idle. In the case when B receives the packet, it will send an acknowledgement message (ACK) back to A, and then forward the packet to the next node in its $\mathbf{FS_A}$ toward the BS. If A does not get the ACK message from B, A retransmits the packet up to a pre-determined number of times ($\boldsymbol{Max_R}$), seven for example; this may happen due to network problems (such as collision) or B's malicious denial. If A does not get the ACK message from B even after the maximum retransmissions $\boldsymbol{Max_R}$, A discards the packet and starts forwarding a next packet in its packet forwarding queue $\boldsymbol{Q_F}$.

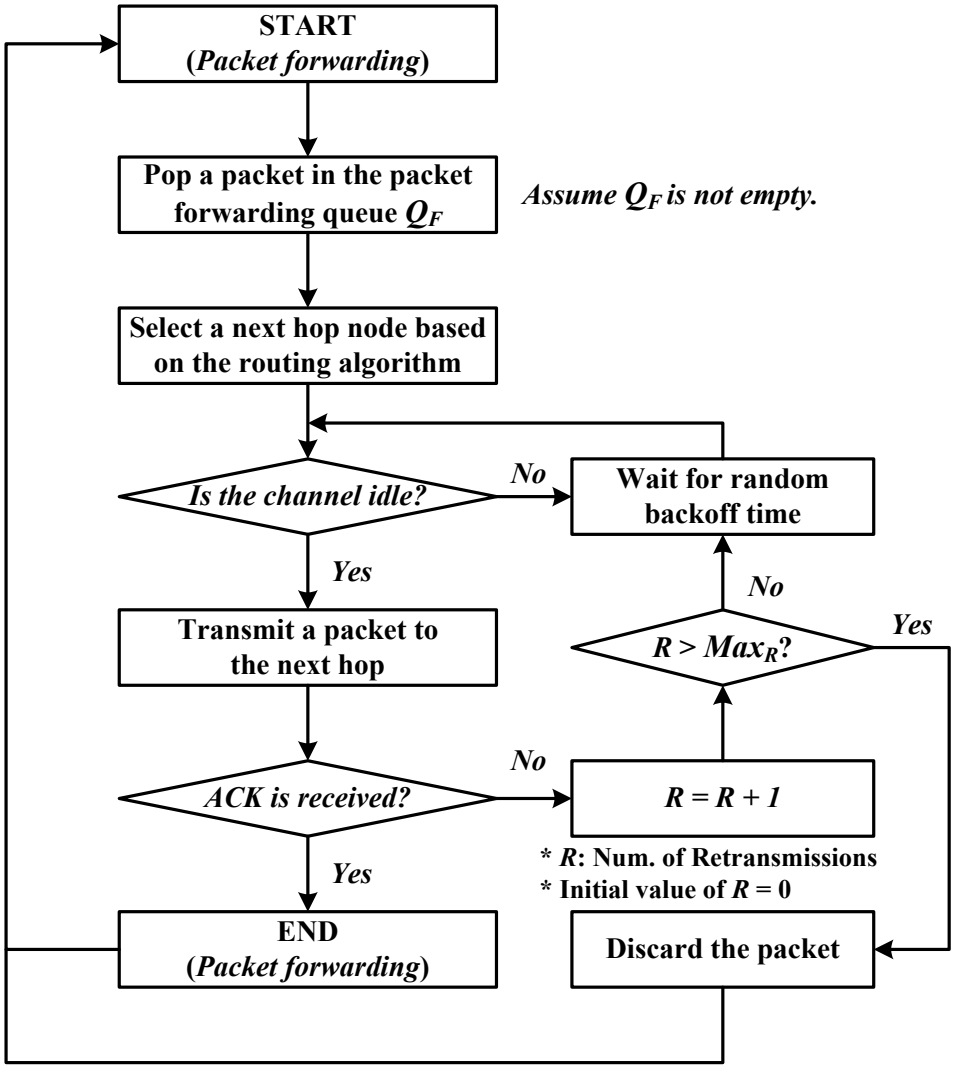


Figure 2.1: Simplified packet forwarding procedure in a WSN.

If B is an insider packet drop attacker, B can simply drop A's packet as follows: (a) B denies receiving A's packet, (b) B receives A's packet, sends ACK back to A, but drops the packet. Both cases will appear to A as natural packet drops due to network problems such as collision, noise, etc. In addition, a retransmission mechanism using ACK cannot defend against B's malicious packet drops. Therefore, we need a special countermeasure to defend against B's malicious packet drop. As we will explain later, if we have a trust mechanism with watchdog, the trust mechanism in A will evaluate the trustworthiness of B based on B's behavioral history to defend against (a) and the direct monitoring mechanism (such as watchdog) in A will monitor B's packet forwarding behavior to check whether or not B forwards A's packet indeed to defend against (b).

2.2 Insider Packet Drop Attacks in WSNs

In WSNs, sensor nodes will generate data packets and send them to the base station (BS) in a multi-hop collaborative fashion due to their limited energy and transmission range. While being routed to the BS, data packets may be lost from collision, congestion, fading, obstacles, or other network problems. The so-called *insider packet drop attacks* refer to a set of attacks where compromised nodes intentionally drop packets [2]. This type of attack has become a serious security threat in WSNs and can cause a serious damage to the network [2, 7, 10]. First, such attackers disguise their malicious behavior behind the aforementioned natural packet loss phenomenon. As a result, it is very difficult to detect them accurately as attackers without false alarms. Second, well-positioned attackers can be on the routing path of many nodes and thus receive many data packets. Thus, they can drop many packets to degrade network

performance or selectively drop critical packets to prevent them from being delivered to their destination (DoS attack). Third, attackers can easily launch these attacks by simply dropping packets without wasting their resources such as energy and memory.

There are several types of packet drop attacks such as blackhole attacks, on-off attacks, and grayhole attacks as described as follows [1, 7, 15].

1) Blackhole attacks: This attacker will drop all the received packets. This attack is the simplest type of packet drop attack, but causes the most serious damage to the network among all types of packet drop attacks. However, the monitoring neighbors can easily capture this attacker, since it consistently drops all their packets.

2) On-off attacks: This attacker will drop all the received packets when it is in attacking mode, and forward all the received packets when the attack is off. It repeats this drop-forward pattern periodically. For example, the attacker drops all received packets during the first 100 seconds (attack on), then forwards all received packets for the next 100 seconds (attack off), and repeats this drop-forward pattern periodically. This attacker can appear suspicious to its monitoring neighbors during its attack period when it acts like blackhole attacker and thus can be detected easily when the attack on period is long or the on-off pattern is discovered.

3) Grayhole attacks (a.k.a., selective forwarding attacks): This attacker will drop some of the received packets, either randomly or selectively. For example, each time the attacker receives a packet, it may decide whether to drop it according to a predetermined attack rate (or drop rate). Another example is that the attacker may only drop packets of specific type or generated by a specific source node.

As described above, it is easy to catch blackhole attackers and there is also some relatively effective means to detect on-off attackers. The grayhole attackers remain as the most difficult to detect because most of the time their behaviors are very similar to those of the normal nodes. Moreover, packet drop attacks have evolved to drop packets intelligently by exploiting internal knowledge about network and security mechanisms to avoid being detected [15].

2.3 Defending Approaches against Insider Packet Drop Attacks

Current defending approaches against insider packet drop attacks are either *avoidance approach* or *detection approach*.

2.3.1 Avoidance Approach

The avoidance approaches focus on how to deliver the packets successfully with the existence of the attackers. A popular way to achieve this is to use multipath routing paths [7, 48, 58, 59]. In [7], the authors pointed out that k disjoint multipath routing can completely defend against selective forwarding attacks with no more than k-1 compromised nodes. However, the multipath routing approach has a couple of drawbacks [4]. First, communication overhead increases significantly as the number of paths increases, and thus it may lead to more collision and interference. As a result, the packet delivery performance of a routing can be dramatically degraded. Second, since this approach cannot catch and discard the attackers, this approach can be compromised if an adversary locates at least one attacker in each routing path. Similarly, a multiple data flow scheme using multiple disjoint topologies was introduced in [5]. In this scheme, a sending

node sends its packets through one or more randomly chosen topologies among the pre-established multiple topologies to mitigate selective forwarding attacks.

2.3.2 Detection Approach

The detection approaches focus on how to quickly and accurately detect the attackers. In general, the attack detection is conducted in two steps: *1) data collection* and *2) evaluation and detection*. Collected data (step 1) is used to evaluate a node and determine whether it is an attacker or not (step 2).

First, data collection mechanisms can be classified into either *direct monitoring mechanisms* or *indirect monitoring mechanisms*. Watchdog [10] is the most popular direct monitoring mechanism by which each evaluating sensor node can observe other nodes' packet transmission behavior. Since it manages only two simple counters (the number of successes and the number of failures), it is well suited to a wireless sensor with a limited memory. This approach has a limitation such that an evaluating node cannot monitor nodes that are out of its overhearing range. To overcome this limitation, the neighbor-based monitoring approaches are introduced as follows. Hai and Huh [17] present a neighbor-based monitoring and detection mechanism using two-hop neighbor knowledge where each exchanges its one-hop neighbors' packet forwarding behavior periodically. In the multi-hop acknowledgement scheme [4], each node in the forwarding path is responsible for detecting attackers. Specifically, some randomly chosen nodes (called ACK nodes) will report ACKs back to the source node (hop by hop) using the same but reversed routing path when they receive a packet. However, these approaches assume that neighbors providing such information are trustful. As a result, these

approaches are vulnerable to false information (e.g., false accusation or bad mouthing) provided by malicious neighbors [10, 15, 23, 42, 43, 44]. Moreover, these approaches introduce network overhead due to periodic information exchange between nodes compared to the direct monitoring approach.

Second, various evaluation and detection approaches are introduced as follows.

The trust mechanisms with the notion of trust in human society have been a promising solution to defend against insider packet drop attacks [1, 6, 12, 15, 22, 25]. A trust model such as the beta trust model, which is a component of trust mechanism, quantitatively measures the trustworthy of nodes based on the data collected by a monitoring mechanism described above and then classifies a node as either trustworthy or untrustworthy according to a predefined threshold. The evaluating node will discard untrustworthy nodes. The attack detection performance such as detection speed or detection accuracy depends on a trust model. It is critical that a trust model should be lightweight in order to be suitable for a wireless sensor.

On the other hand, in the incentive-based approach [61, 62], nodes are given virtual credits initially and they need to pay a certain amount of credits to forward their packets toward the destination. To earn credits, a node must forward other nodes' packets cooperatively. This approach forces a node to be collaborative to be participating in packet forwarding activities. In addition, sophisticated intrusion detection approaches based on data mining technique are introduced for WSNs [63, 64]. While these approaches have high detection accuracy, these approaches introduce high complexity and thus do not fit a small, inexpensive wireless sensor with limited resources [27].

In this dissertation, we focus on advancing the state-of-the-art trust mechanisms with a direct monitoring mechanism watchdog for WSNs.

2.4 Watchdog

Marti et al [10] introduced a monitoring mechanism known as watchdog to identify misbehaving nodes in wireless ad hoc networks. In their approach, each sensor node has its own watchdog that monitors and records its one hop neighbors' behaviors such as packet transmission. When a sending node S sends a packet to its neighbor node T, the watchdog in S verifies whether T forwards the packet toward the BS or not by using the sensor's overhearing ability within its transceiver range.

In this mechanism, S stores all recently sent packets in its buffer, and compares each packet with the overheard packet to see whether there is a match. If yes, it means that the packet is forwarded by T and S will remove the packet from the buffer. If a packet remains in the buffer for a period longer than a pre-determined time, the watchdog considers that T fails to forward the packet and will increase its failure tally for T. If a neighbor's failure tally exceeds a certain threshold, T will be considered as a misbehaving node by S. Watchdog works similarly with trust mechanism in that trust model evaluates each sensor's trustworthiness based on the past behaviors in much sophisticated ways.

Although watchdog is a popular mechanism for direct neighbor monitoring, unfortunately it cannot completely monitor its neighbor's packet forwarding behaviors. In [10], Marti et al pointed out that watchdog is limited by not being able to detect a misbehaving node in the presence of the following cases. We briefly examine each case using the path S→A→B→C→BS as shown in Figure 2.2 as an example.

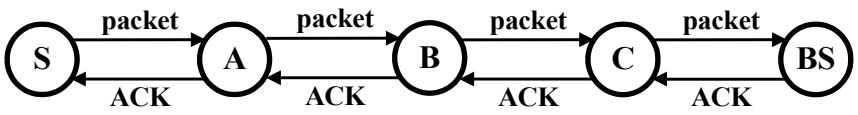


Figure 2.2: A routing path S→A→B→C→BS.

1) Ambiguous collision: Consider the situation that A forwards a packet to B, and then starts to overhear whether B will forward the packet to C. However, when B forwards to C, A may not overhear this transmission if other neighbors (such as S) send packets to A at the same time. This collision may mislead A to conclude that B is malicious, which may not be correct.

2) Receiver collision: Similar to the above case, a collision may also occur at the receiver side C resulting in C not receiving the packet correctly. A can only overhear that B has forwarded the packet, but A cannot tell whether C has received. When this happens, (malicious) node B can intentionally skip retransmissions or (malicious) node C can generate collision on purpose to avoid receiving the packet.

3) Limited transmission power: If B adjusts its transmission power such that A can overhear but C cannot receive, B can not only drop packets but also increase its trustworthiness (to node A). In geographic routings where every node knows the positions of itself and its neighbors, B can easily launch this attack as follows. B selects a node C from its FS such that *dist* (B, C) > *dist* (B, A) where *dist* (*i*, *j*) is a distance between nodes *i* and *j*, and then transmits its packet by maliciously adjusting power such that the packet can reach A but cannot reach C.

4) False misbehavior: This case happens when a malicious node intentionally reports that other nodes are misbehaving. For example, A may report B is dropping

packets although B is not. Then, A's neighbor such as node S, which cannot directly communicate (and thus monitor) B, will consider B malicious.

5) Collusion: Multiple colluding attackers can launch more sophisticated attacks. For example, two malicious colluding nodes A and B can completely deceive S if A forwards all packets from S to B, but B drops all the packets. Since S cannot overhear B's drops, S will not consider A and B malicious.

6) Partial dropping: Instead of dropping all packets, B can drop only some packets such that the failure tally will not exceed the detection threshold of A's watchdog. This is similar to the grayhole attacks.

Despite its many known limitations, the watchdog has been widely used as a direct monitoring mechanism that enables a sender to monitor a receiver's packet forwarding behavior to its packets by resolving the limitations of the ACK mechanism, as described in Section 2.1. In this dissertation, to avoid any confusion, we consider that watchdog is a component in the trust mechanism and it is responsible for node behavior monitoring. The limitations of watchdog will be the limitations of trust mechanisms. Although we do not enhance all limitations of watchdog introduced above in this dissertation, we study how to reduce the damage caused by the inside packet drop attackers in the presence of current trust mechanisms with the watchdog.

2.5 Trust mechanism

2.5.1 Notion of trust

Gambetta built a framework where trust can be measured mathematically by defining trust as the subjective probability by which one individual expects that another

individual will perform a given action on which its welfare depends [46]. In computer networks, Sun et al [12, 23] introduced trust as a belief -- one entity believes that the other entity will act in a certain way. Yu et al [15] defined trust as a subjective opinion in the reliability of other entities or functions, including veracity of data, connectivity of path, processing capability of node, and availability of service, etc.

2.5.2 Three Working Stages of Trust Mechanism

A trust mechanism defines a trust value (or trustworthiness) for each sensor node, and how each node measures the trustworthiness of its neighbors. It detects insider packet drop attacks in the three stages: neighbor behavior monitoring, trust measurement, and attacker detection. We now elaborate each of these stages.

1) Neighbor behavior monitoring: Each node monitors and records its neighbors' behaviors such as packet forwarding. Watchdog is a popular monitoring mechanism used in this stage. The confidence of the trustworthiness evaluation depends on how much data a sensor collects and how reliable such data are. That is, if the collected data are not trustful, the entire trust mechanism cannot be trustful, either.

2) Trust measurement: Trust model defines how to measure the trustworthiness of a sensor node based on the data collected in the previous stage. Yu et al [15] introduced several representative approaches to build the trust model, which include Bayesian approach, Entropy approach, Fuzzy approach, etc. The trust value of a node may be different when we use different trust models. For example, when a node is observed to forward the packet s times and drops the packet f times, the beta trust model [6] will assign trust value T ($0 \leq T \leq 1$) to this node using the following formula

$$T = \frac{s+1}{s+f+2}. \tag{2.1}$$

Meanwhile, entropy trust model [12] will assign trust value T ($-1 \le T \le 1$) to this node the following formula

$$T = \begin{cases} 1 - H(p), for\ 0.5 \le p \le 1; \\ H(p) - 1, for\ 0 \le p < 0.5 \end{cases} \tag{2.2}$$

where $H(p) = -p\log_2 p - (1-p)\log_2(1-p)$ and $p = \frac{s+1}{s+f+2}$.

We will explain trust models in detail in Section 2.5.3.

3) Attack detection: By comparing the measured trust value with a pre-determined threshold $\boldsymbol{\Theta_T}$, a node can decide whether its neighbor is trustworthy. If the neighbor's trust value is less than $\boldsymbol{\Theta_T}$, it will be considered as an inside packet drop attacker and thus it will be discarded by the evaluating node. Depending on the network's trust mechanism, the detection of inside packet drop attackers may or may not be broadcast to the rest of the nodes in the network. In our dissertation, we assume that the decision will not be broadcast for simplicity.

2.5.3 Trust Models

We describe in detail two representative trust models based on Bayesian and Entropy approaches which we are going to compare with our proposed trust model. In addition, we briefly introduce a couple of other approaches to model trust.

1) Beta trust model based on Bayesian approach

Josang [6] proposed a Bayesian approach-based beta trust model, which has since been widely used in most computer networks due to its simplicity. The beta trust model is

designed by using a mathematical fact [3] that the posterior probability p of a binary event indexed by α and β can be represented by beta distribution using gamma function Γ as

$$\begin{aligned} f(p \mid \alpha, \beta) &= \frac{p^{\alpha-1}(1-p)^{\beta-1}}{\int_0^1 u^{\alpha-1}(1-u)^{\beta-1}\,du} \\ &= \frac{p^{\alpha-1}(1-p)^{\beta-1}}{B(\alpha,\beta)} \\ &= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\,p^{\alpha-1}(1-p)^{\beta-1} \end{aligned} \tag{2.3}$$

where beta function B (α, β) = $\int_0^1 u^{\alpha-1}(1-u)^{\beta-1}\,du = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$, gamma function Γ (n) = (n-1)! for a positive integer n, $0 \le p \le 1$, $\alpha > 0$, $\beta > 0$, and the following two restrictions: $p \neq 0$ if $\alpha < 1$, and $p \neq 1$ if $\beta < 1$. The expectation value of (2.3) is given by

$$\begin{aligned} E[f(p \mid \alpha, \beta)] &= \int_0^1 p \cdot f(p \mid \alpha, \beta)\,dp \\ &= \int_0^1 p\left(\frac{p^{\alpha-1}(1-p)^{\beta-1}}{\int_0^1 u^{\alpha-1}(1-u)^{\beta-1}\,du}\right)dp \\ &= \frac{\int_0^1 p^{(\alpha+1)-1}(1-p)^{\beta-1}\,dp}{B(\alpha,\beta)} \\ &= \frac{\Gamma(\alpha+\beta)\Gamma(\alpha+1)}{\Gamma(\alpha+\beta+1)\Gamma(\alpha)} \\ &= \frac{\alpha!(\alpha+\beta-1)!}{(\alpha+\beta)!(\alpha-1)!} \\ &= \frac{\alpha}{\alpha+\beta} \end{aligned} \tag{2.4}$$

When we want to evaluate an entity's trust based on observations of whether the entity performs an action successfully or not, let s and f the number of successes and failures, respectively. Beta trust model uses the expectation of p with $\alpha = s+1$ and $\beta = f+1$ to measure the trust value of the entity, that is

$$T_{Beta} = E[f(p \mid s+1, f+1)] = \frac{s+1}{s+f+2}. \tag{2.5}$$

For trust evaluation, the beta trust model just needs to keep s and f of an entity, and use the simple equation (2.5) at a certain time t. This simplicity makes the beta trust model very suitable for WSNs, and many researchers have used this model as a part of their trust mechanisms or trust-based applications [23, 29, 30, 44, 53, 70].

2) Entropy trust model based on Information Theory

Sun et al [12] proposed a unique trust model using the notion of entropy for wireless mobile ad hoc network, where they measure the uncertainty in belief between two entities, *subject* and *object*. In information theory, entropy is a measure for the uncertainty of a random variable [51]. Given a binary random variable X whose value is 1 with probability p, entropy $H(X)$ is defined as

$$H(X) = -p\log_2 p - (1-p)\log_2(1-p) \overset{def}{=} H(p). \tag{2.6}$$

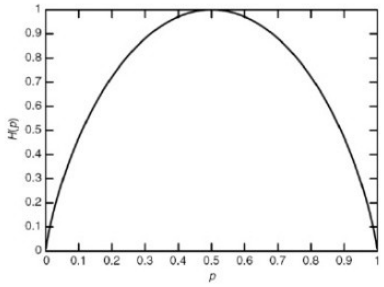


Figure.2.3: Entropy function $H(p)$. $H(0) = H(1) = 0$ and $H(0.5) = 1$.

As Figure 2.3 shows, entropy $H(X)$ reaches its maximum at $p = 0.5$, when the value of X is equally likely to be 0 or 1, exhibiting the largest uncertainty; and there is no uncertainty when $p = 0$ or 1. Using $H(p)$, a trust can be defined as

$$T_{Entr} = \begin{cases} 1 - H(p), for\ 0.5 \le p \le 1; \\ H(p) - 1, for\ 0 \le p < 0.5. \end{cases} \quad (2.7)$$

where the value of p is obtained using the same beta distribution function as that in the beta trust model. In this trust model, trust value reaches to its maximum (that is, 1) when p = 1 (maximum uncertainty) and to its minimum (that is, -1) when p = 0 (maximum uncertainty); trust value becomes 0 when p = 0.5 (minimum uncertainty). Therefore, we can consider entropy trust model as a combination of Bayesian and Entropy approaches.

Note that in (2.5), the trust value is between 0 and 1. But the trust value in (2.7) is between -1 and 1. To have a non-negative trust value between 0 and 1, we define

$$T^*_{Entr} = \frac{1 + T_{Entr}}{2}. \quad (2.8)$$

3) Other approaches

Similar to the entropy-based approach, Fuzzy-based approaches have been introduced to WSNs to handle the uncertainty in trust management [15, 65]. In these approaches, based on the predefined set of rules and obtained data about a node, a fuzzy inference engine simulates humans' decision-making processes to determine the trustworthiness of a given node. However, these approaches have high memory and computation cost and may not be appropriate for WSNs applications.

Game theory-based approaches have also been introduced in wireless ad hoc networks to discard uncooperative nodes and achieve cooperation among nodes in the network [15, 66]. Jaramillo et al [66] presented a reputation mechanism evaluating the trustworthiness of nodes interacting based on a Prisoner's dilemma model. However, Yu et al [15] pointed out that game theory is not a suitable approach for WSNs because the

game theory is not a predictable tool for nodes' behaviors, and moreover the characteristic of one-way transmission in general WSNs does not satisfy the bidirectional behaviors, which is a requirement for game theory.

2.6 Trust-based Routing (Trust-aware Routing)

Here we explain how the trust-based routing defends against insider packet drop attacks by using the Greedy Perimeter Stateless Routing (GPSR) [8], which is a representative geographic routing used in wireless ad hoc and sensor networks [22, 67, 68], and a simple trust-based routing based on the GPSR.

2.6.1 Greedy Perimeter Stateless Routing (GPSR)

1) Greedy forwarding

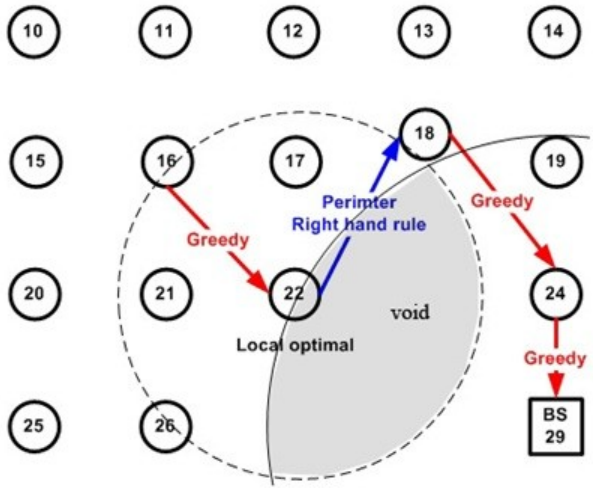


Figure 2.4: Greedy forwarding and perimeter forwarding in the GPSR. Greedy forwarding finds a neighbor closest to the BS. When greedy forwarding fails in finding such a neighbor (e.g., when node 22 meets a *void* or a *hole* in the figure), perimeter forwarding is used to find a path to the BS.

The sending node S forwards a packet to a node nearest to the destination using its one-hop neighbors' geographic information. Let $d(n_i, n_j)$ denote the Euclidean distance between two nodes n_i and n_j. Then, S selects node n^* as the next hop where n^* has the minimum distance $d(n^*, D)$ to destination D among S and all its neighbors. For example, in the small WSN shown in Figure 2.4, node 16 will choose node 22 as the next hop because node 22 is closer to the destination node 29 (called base station) than any of 16's other neighbors (nodes 10, 11, 12, 15, 17, 20, and 21) and node 16 itself. However, this greedy forwarding may fail when S becomes the local optimal where S has no neighbors closer to the destination than S itself. Node 22 in Figure 2.4 is one of such local optimal. In this case, we say that S meets a *void* (or a *hole*) as shown in the shaded region in Figure 2.4. When this occurs, a perimeter forwarding method below will be used to find an alternative node to the destination.

2) Perimeter forwarding (when greedy forwarding failed)

When the sending node S meets a *void*, it can forward the packet to a node on the perimeter of a planarized graph [8], such as Relative Neighborhood Graph (RNG) or Gabriel Graph (GG), based on the right hand rule to the clockwise direction. Perimeter is a sequence of edges traversed following the right-hand rule. Planar graph is a graph where no two edges cross and it can be used to enforce the right hand rule. The complexity of generating RNG or GG takes $O(deg^2)$ time at each node where *deg* is the node's degree in the radio graph. For example, in Figure 2.4, nodes 22, 18, and 24 form a planarized graph and node 18 along the perimeter can be selected to receive the messages from node 22.

2.6.2 Trust-based GPSR

In order to enhance the security and reliability of routing solutions in the network, the notion of trust has been considered. The dotted line 16 → M22 → 28 → BS29 in Figure 2.5 is the routing path by GPSR. When nodes M22 and M23 become inside packet drop attackers and start performing blackhole attack, all their received packets will be dropped. Using a monitoring mechanism such as watchdog, their neighbors will notice the drop of the packets and reduce the trust value of nodes M22 and M23. When their trust value goes below a pre-defined threshold $\boldsymbol{\Theta_T}$, a trust-based GPSR as simple as forwarding only to node whose trust value is above the threshold $\boldsymbol{\Theta_T}$ will avoid nodes M22 and M23 and find the solid line 16 → 17 → 18 → 24 → BS29 as a trusted routing path.

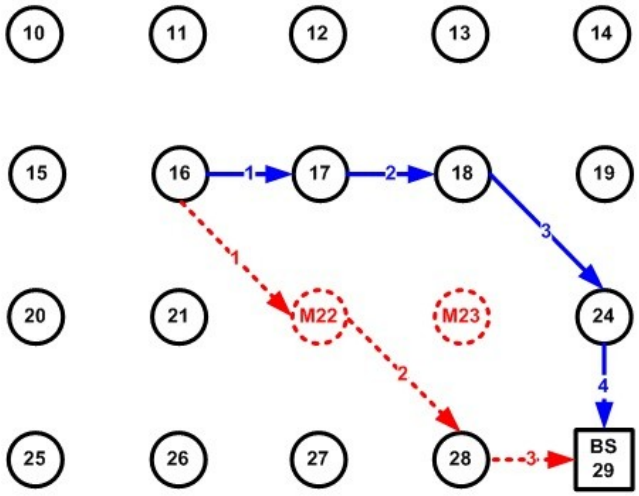


Figure 2.5: GPSR (Dotted lines) and Trust-based GPSR (Solid lines). In the presence of inside packet drop attackers (nodes M22 and M23), the trust-based GPSR finds a trustful routing path (16→17→18→24→BS29) to the BS by avoiding malicious nodes.

Many researchers [12, 16, 22, 25, 71, 72, 73] have showed that trust-based routing approaches can gracefully mitigate insider packet drop attacks by building trusted paths to the destination. Moreover, they showed that trust-based routing improves the packet's successful delivery rate under insider packet drop attacks over routing algorithms that do not consider trust. Clearly, the effectiveness of these trust-based routing protocols is based on the underlying trust model. A good trust model will help the routing algorithm to quickly and accurately identify insider packet drop attackers and find alternate routes to avoid them.

In this dissertation, we will use a geographic routing protocol that is similar with GPSR, which is Geographic Routing Protocol (GRP) provided by the OPNET Modeler.

Chapter 3

Hybrid Trust Model against Insider Packet Drop Attacks

3.1 Overview

Inside attackers are also legitimate members of the network and they will know the trust mechanism being used. As we will elaborate later, it is trivial for attackers to launch insider packet drop attacks without being detected by the existing trust mechanisms. Meanwhile, inside attackers can continue to drop packets and cause damage to the network performance before they are detected and eliminated from the network. Therefore, quick and accurate detection of inside packet drop attackers is critical to minimize the damage to the network, especially for networks that carry out mission-critical applications Clearly, any characteristics that can distinguish inside packet drop attackers from other normal nodes will help in evaluating a node's trustworthiness and thus improves the detection effectiveness of trust mechanisms.

In this chapter, we propose *consecutive packet drops* as one of such characteristics because of the following three reasons. First, in many applications, inside packet drop attackers have to drop packets consecutively to achieve their goal and this has been demonstrated in the previous studies [1, 15, 25]. Second, a large number of consecutive drops are unlikely to happen naturally at a normal node. Third, when a normal node drops packets consecutively due to network problem, although it is not an attacker, it may be advantageous to eliminate such node from the routing paths to improve network performance. Basically, the first two reasons indicate that consecutive packet drops can distinguish the behavior between attackers and normal nodes; the last reason argues that a

false alarm (that is, misclassifying a normal node as an attacker) may not cause much damage to the network performance.

Current trust models, including the popular Beta trust model [6] and Entropy trust model [12], only consider the total number of packet drops and do not address the distribution of these drops. As we will show later in this chapter, such models cannot be effective in detecting inside attackers. To the best of our knowledge, this is the first attempt to build trust models based on packet dropping properties other than the total number. Our trust model takes consecutive packet drops into consideration and can be seamlessly integrated into existing models.

To show that evaluating the trust value of a node based on its consecutive packet drops can effectively improve the detection speed and accuracy, we propose a new trust model based on the concept of consecutive drops, and then build a hybrid trust model that can adaptively choose between this new trust model and an existing trust model. We conduct extensive simulations using a commercially available network simulator, OPNET Modeler [21], to demonstrate that our model outperforms the beta trust model for all types of inside packet drop attacks not only in terms of detection speed and accuracy as it is designed for, but also in terms of other important network performance metrics such as packet delivery rate, routing reliability, and energy-efficiency.

The rest of this chapter is organized as follows. In Section 3.2, we give the motivation for considering consecutive packet drops in evaluating trust in order to quickly and accurately detect inside packet drop attackers. In Section 3.3, we elaborate our hybrid trust model based on consecutive drops. In Section 3.4, we evaluate the performance of our model. Finally, we summarize in Section 3.5.

3.2 Problem Description

3.2.1 Consecutive Packet Drops

Assuming that inside packet drop attackers reside in the network, detecting them as soon as possible (*early detection*) is critical to minimize the damage they may cause to the network, because undetected attackers repeatedly drop packets and thus degrade the network performance. We propose to use *consecutive drops* (or *consecutive failures*) for the early detection of inside packet drop attackers. Evaluating trust based on consecutive drops improves the early detection ability of a trust model because of the following two reasons.

First, an inside packet drop attacker drops packets consecutively to cause serious damage to the network. In this dissertation, we do not consider an attacker that cannot cause significant damage to the network. According to most known packet drop attack models, the attacker achieves this goal either by dropping as many packets as possible to significantly degrade the packet delivery performance or by preventing some critical data packets from reaching the destination (DoS attack) [11, 14, 49]. To do so, an attacker will generate frequent or long consecutive drops. If an attacker does not drop packets consecutively, the maximum damage that it can cause will be very limited. For example, in the former case, an attacker cannot drop more than 50% of packets that it receives without dropping packets consecutively. Meanwhile, the attacker can drop up to 100% of receive packets if it drops packets consecutively. Thus, if we consider consecutive drops in evaluating the trustworthiness of such attackers, we can better defend against them. For the latter case, consider a WSN deployed in a territory for intruder detection. With the help of insiders that perform the DoS attack, an intruder will be able to enter the territory

from the area monitored by victim nodes. If a malicious insider does not consecutively drop critical data packets with the information of the intruder, some of the critical packets will reach the BS, and thus the BS will detect the intruder. In this intruder detection problem, consecutive drops are more critical than the former case (network performance degradation). Similar practical examples can be found in many WSN applications such as fire detection, enemy tracking, battlefield surveillance, and disaster prevention [19, 24].

Second, a large number of consecutive drops are unlikely to happen at a normally functioning node in the network. We can probabilistically measure the degree of abnormality of a node generating consecutive drops based on the number of consecutive drops. That is, as the size of consecutive drops n grows, the belief that the node generating the n consecutive drops is an abnormal node such as a packet drop attacker or a faulty node also grows. Let $P[f]$ be the probability that a packet is dropped at a normally functioning node. Then, the probability that a packet is forwarded by the node successfully will be $P[s] = 1 - P[f]$. If we observe 10 consecutive drops at the node, then the probability that the 10 drops occur consecutively is $P[f]^{10}$. When we assume $P[f] < P[s]$ in a well-designed WSN, even if $P[f]$ is 0.5, $P[f]^{10} \approx 0.000977$, which means that it is very unlikely that 10 consecutive drops are happening. As n grows, the probability that the n consecutive drops happen ($P[f]^n$) decreases exponentially, and the probability that the n consecutive drops would not happen ($1 - P[f]^n$) also grows. If $n > m > 0$, we believe that n consecutive drops are more abnormal and suspicious to inside packet drop attacks than m consecutive drops. Based on this reasoning, we will show how we can penalize a node based on the number of consecutive drops in Section 3.3.

Meanwhile, using consecutive drops for detecting packet drop attack may

generate a false alarm. In practice, a node (non-attacker) may also drop many packets consecutively due to fading or obstacles. Detecting such node as an attacker is a false alarm. However, if a node (non-attacker) generates a large number of consecutive drops, it is not functioning normally, and we should consider such node untrustworthy to ensure the quality of packet delivery performance of a routing. In fact, we show that avoiding such problematic nodes quickly reduces the packet drop rate of a routing in Section 3.4.

3.2.2 Existing Trust Models and Consecutive Drops

In this section, we use an example to show what an inside packet drop attacker can do against the existing trust models that do not consider consecutive drops.

Consider the two observations that contain 10 successes and 10 failures such as ffssfsfsfsfffsssfsfs and sssssssssfffffffff. Both trust models will treat them equally although the latter sequence looks more suspicious to the insider packet drop attack due to the last 10 consecutive drops according to the above reasoning. This is also supported by a practical assumption that inside attackers start attacks after they develop high trust (e.g., the first 10 successes in the latter observation) to avoid being easily detected by the trust mechanism [20].

We now show how the beta and entropy trust models may fail to quickly detect a naive blackhole attacker through a simple analysis. Suppose that a node' trust value is approximately 1 (the node is very trustful) after it successfully forwarded 1,000 packets (that is, s = 1,000), and then the node starts dropping packets. As the number of consecutive drops n goes from 1 to 30, Figure 3.1 shows how their trust values T drop; Trust values in the beta trust model and the entropy trust model are calculated by (2.5)

and (2.8), respectively. Surprisingly, after 30 consecutive drops, the trust values in the beta trust model and the entropy trust model are 0.969 and 0.902, respectively, which means the blackhole attacker will not be detected unless we use very high trust thresholds. However, we cannot simply use a very high trust threshold, since it may misclassify many good nodes as untrustworthy nodes. Thus, we need to build a new trust model that considers consecutive drops such that it will significantly penalize a node's trust value when it drops consecutive packets.

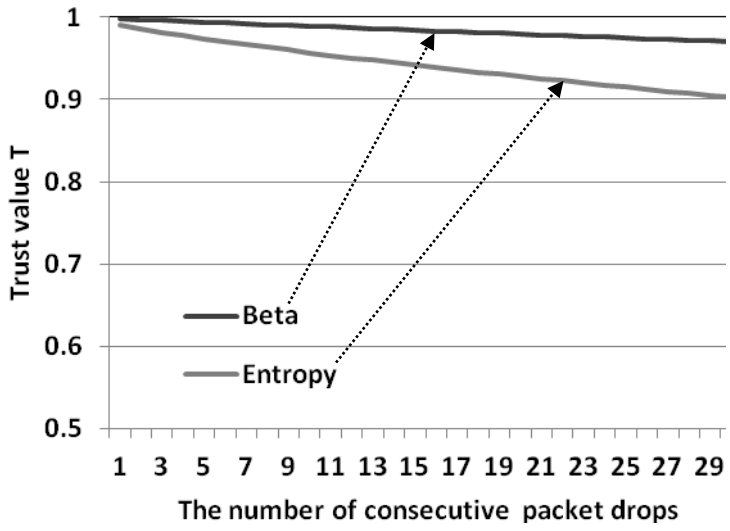


Figure 3.1: Beta and entropy trust models in the presence of consecutive packet drops. Even after 30 consecutive packet drops, the trust values in the beta trust model and the entropy trust model are still very high (i.e., $T_{Beta} \approx 0.969$ and $T^{*}_{Entr} \approx 0.902$).

3.3 Proposed Trust Model

3.3.1 Hybrid Trust Model

To show that trust evaluation using consecutive drops can effectively improve the early detection ability of a trust model, we propose a hybrid trust model that adaptively uses the popular beta trust model and our new trust model based on consecutive drops. As shown in Figure 3.2, on each report of the behavior of a node B (that is, node A wants to evaluate the trust of node B), our hybrid trust model in node A works as follows:

• If the new behavior is a success, use the beta trust model;

• If the new behavior is a failure, use the new trust model based on the number of consecutive drops (or consecutive failures).

For a single failure, our new trust model will work like the beta trust model. For consecutive failures, our new trust model evaluates B's trustworthiness based on the number of consecutive failures.

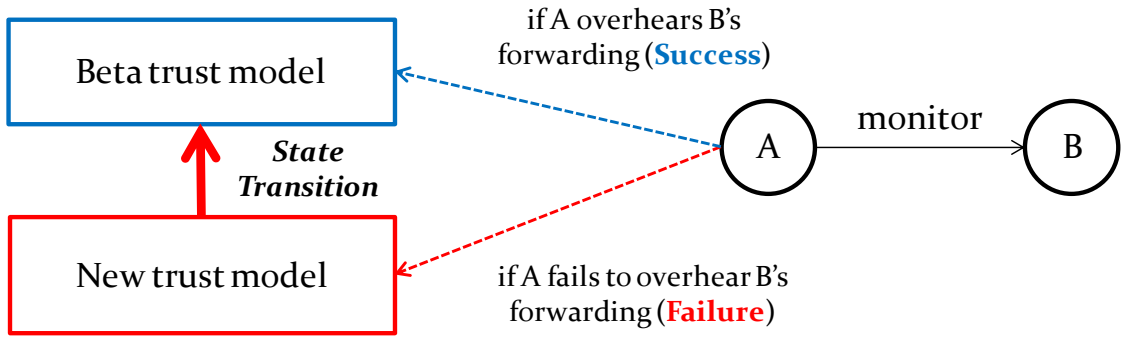


Figure 3.2: Hybrid trust model in node A. For each observation on node B's behavior, when A observes that B forwards A's packet successfully, A increases B's trust value by using the beta trust model. Otherwise, A decreases B's trust value by using the new trust model based on the number of consecutive drops.

3.3.2 New Trust Model

3.3.2.1 Properties against Insider Packet Drop Attacks

We demand that the new trust model has the following two properties:

• *Property 1*: As the number of consecutive failures grows, we increase the trust-decreasing rate. In other words, while we observe consecutive failures, we give a larger penalty for a new failure than the previous failures, because as the size of consecutive failures grows, our confidence that the consecutive failures are occurring due to attacks also grows. This property not only helps to quickly detect an attacker dropping packets consecutively, but also forces an intelligent inside attacker to stop launching many consecutive drops for fear of being caught by the trust model.

• *Property 2*: Given two nodes B and C such that node A trusts B more than C based on the same number of their past behaviors ($T_A[B] > T_A[C]$), if both B and C are generating the same number of consecutive failures, A decreases on C's trust more than on B's trust. For example, consider the following case. B behaved well 40 times and behaved badly 20 times ($s_B = 40$ and $f_B = 20$), and node C behaved well 20 times and behaved badly 40 times ($s_C = 20$ and $f_C = 40$). Now both B and C are misbehaving five times consecutively. Intuitively, it is reasonable to lower C's trust further than B, because C's past behaviors are worse than node B. This property will work against grayhole attacks and on-off attacks.

3.3.2.2 Design of New Trust Function

We design a new trust function based on the number of consecutive failures. Every trust model decreases a certain amount of trust value when a failure occurs. We

consider the decreased amount of trust value as a *penalty* for the failure. We first design a penalty function $PT(n)$ that determines how much we should lower the trust value of a node that is generating n consecutive failures. By using $PT(n)$, given n consecutive failures, new trust function $T(n)$ is formulated as

$$T(n) = T(0) - PT(n) \tag{3.1}$$

where $T(0)$ is the initial trust value of the evaluated node generating n consecutive failures at the evaluating node.

We now explain how to determine the amount of penalty for n consecutive failures. Consider n consecutive failures $f_1 f_2 \ldots f_i \ldots f_n$ whose i-th failure occurs at time $t = i$ ($1 \leq i \leq n$). When $t = 1$, there is only one failure f_1 and we assign α as a penalty to the single failure f_1; Thus, $PT(1) = \alpha$. When $t = 2$, another failure f_2 occurs. Therefore, we assign α to the single failure f_2. In addition, since we observe double failures $f_1 f_2$ at $t = 2$, we give an extra penalty to $f_1 f_2$. Let $P[f]$ be the probability that a packet is dropped at the evaluated node. The probability that two consecutive failures occur is $P[f]^2$ and our confidence that they should not occur is $1–P[f]^2$. Since we give α as the penalty to a single failure when we have confidence of $1–P[f]$, by multiplying α by the ratio of confidence, we can get the penalty to the double failure as $\alpha \times (1–P[f]^2)/(1-P[f])$. Then, the total penalty to the two consecutive failures $PT(2)$ is $2\alpha + \alpha \times (1–P[f]^2)/(1-P[f])$ where 2α is the penalty for two single failures and $\alpha \times (1–P[f]^2)/(1-P[f])$ is the penalty for one double failure.

Table 3.1: Unit failure and unit penalty used in the penalty function $PT(n)$. Unit penalty for each unit failure is obtained probabilistically by multiplying the base penalty α by the confidence ratio of the unit failures.

Unit failure	Probability	Confidence	Confidence ratio	Unit penalty
Single f	$P[f]$	$1\text{-}P[f]$	1	α
Double ff	$P[f]^2$	$1\text{-}P[f]^2$	$(1\text{-}P[f]^2)/(1\text{-}P[f])$	$\alpha\times(1\text{-}P[f]^2)/(1\text{-}P[f])$
Triple fff	$P[f]^3$	$1\text{-}P[f]^3$	$(1\text{-}P[f]^3)/(1\text{-}P[f])$	$\alpha\times(1\text{-}P[f]^3)/(1\text{-}P[f])$
n-tuple f…f	$P[f]^n$	$1\text{-}P[f]^n$	$(1\text{-}P[f]^n)/(1\text{-}P[f])$	$\alpha\times(1\text{-}P[f]^n)/(1\text{-}P[f])$

By this approach, to get $PT(n)$, we first find all *unit failures* such as single failure, double failure, triple failure, and n-tuple failure within n consecutive failures. Given n consecutive failures, there are n single failures, n-1 double failures, n-2 triple failures… and one n-tuple failure. Next, as shown in Table 3.1, we calculate *unit penalty* for each unit failure probabilistically by multiplying the base penalty α by the confidence ratio of the unit failures. By this manner, we formulate $PT(n)$ as

$$\begin{aligned} PT(n) &= n\times\alpha + (n-1)\times\alpha\times\frac{1-P[f]^2}{1-P[f]} + (n-2)\times\alpha\times\frac{1-P[f]^3}{1-P[f]} + \\ &\quad ... + 2\times\alpha\times\frac{1-P[f]^{n-1}}{1-P[f]} + 1\times\alpha\times\frac{1-P[f]^n}{1-P[f]} \\ &= \sum_{i=1}^{n}\alpha\times(n-i+1)\times\frac{1-P[f]^i}{1-P[f]} \end{aligned} \tag{3.2}$$

where α is a base penalty for a single failure.

By using (3.1) and (3.2), we get $T(n)$ as

$$T(n) = T(0) - \sum_{i=1}^{n}\alpha\times(n-i+1)\times\frac{1-P[f]^i}{1-P[f]} \tag{3.3}$$

where $P[f] = 1 - T(0)$ since $T(0)$ is the expectation value of $P[s]$ when $n = 0$ in the beta trust model.

To reduce the computation complexity, we have

Theorem 3.1. The trust function *T*(n) defined in (3.3) satisfies

$$T(n) = T(n-1) - PT_R(n); \tag{3.4}$$

$$PT_R(n) = P[f]PT_R(n-1) + \alpha n. \tag{3.5}$$

Proof. From (3.2), we know

$$\begin{aligned} PT(n-1) &= (n-1)\times\alpha + (n-2)\times\alpha\times\frac{1-P[f]^2}{1-P[f]} + \\ &\quad ... + 2\times\alpha\times\frac{1-P[f]^{n-2}}{1-P[f]} + 1\times\alpha\times\frac{1-P[f]^{n-1}}{1-P[f]} \\ &= \sum_{i=1}^{n-1}\alpha\times(n-i)\times\frac{1-P[f]^i}{1-P[f]}. \end{aligned} \tag{3.6}$$

From (3.3), by using (3.6), we have

$$\begin{aligned} T(n) &= T(0) - \sum_{i=1}^{n}\alpha\times(n-i+1)\times\frac{1-P[f]^i}{1-P[f]} \\ &= T(0) - \left\{\sum_{i=1}^{n-1}\alpha\times(n-i)\times\frac{1-P[f]^i}{1-P[f]} + \alpha\times\sum_{i=1}^{n}\frac{1-P[f]^i}{1-P[f]}\right\} \\ &= \left\{T(0) - \sum_{i=1}^{n-1}\alpha\times(n-i)\times\frac{1-P[f]^i}{1-P[f]}\right\} - \alpha\times\sum_{i=1}^{n}\frac{1-P[f]^i}{1-P[f]} \\ &= T(n-1) - \alpha\times\sum_{i=1}^{n}\frac{1-P[f]^i}{1-P[f]}. \end{aligned} \tag{3.7}$$

Next, let $PT_R(n) = \alpha\times\sum_{i=1}^{n}\frac{1-P[f]^i}{1-P[f]}$. Then,

$$
\begin{aligned}
PT_R(n) &= \alpha \times \sum_{i=1}^{n} \frac{1-P[f]^i}{1-P[f]} \\
&= \alpha \left\{ 1 + \frac{1-P[f]^2}{1-P[f]} + \frac{1-P[f]^3}{1-P[f]} + \cdots + \frac{1-P[f]^n}{1-P[f]} \right\} \\
&= \alpha \begin{Bmatrix} 1 + (1+P[f]) + (1+P[f]+P[f]^2) + \cdots \\ + (1+P[f]+\cdots+P[f]^{n-1}) \end{Bmatrix} \\
&= \alpha \begin{bmatrix} n + P[f]\{1 + (1+P[f]) + (1+P[f]+P[f]^2)\cdots \\ + (1+P[f]+\cdots+P[f]^{n-2})\} \end{bmatrix} \\
&= \alpha \left\{ n + P[f] \left(\sum_{i=1}^{n-1} \frac{1-P[f]^i}{1-P[f]} \right) \right\} \\
&= P[f] PT_R(n-1) + \alpha n.
\end{aligned}
\tag{3.8}
$$

■

3.3.2.3 Base Penalty α

Next, we need to determine the value of the base penalty α. For a single failure (n=1), we assign the amount of penalty that the beta trust model gives for a single failure by which our new trust function works the same as the beta trust model. This approach helps to make it easy to design hybrid trust model as we explained in Section 3.3.1. Thus, given the number of successes s and the number of failures f,

$$
\begin{aligned}
\alpha_{n=1} &= T_{Beta}(s, f) - T_{Beta}(s, f+1) \\
&= \frac{s+1}{s+f+2} - \frac{s+1}{s+f+3}. \\
&= \frac{(s+1)}{(s+f+2)(s+f+3)}.
\end{aligned}
\tag{3.9}
$$

Meanwhile, as we have explained, for consecutive failures ($n \geq 2$), we use our new trust model that penalizes the node generating the consecutive failures. However, since $\alpha_{n=1}$ in (3.9) is not inversely proportional to a node's initial trust value $T(0)$ and thus

it does not satisfy *Property 2*. This is because of the design of the beta trust model. To satisfies Property 2, we use a different base penalty for $n \geq 2$ by dividing α by the initial trust value $T(0)$ as

$$\alpha_{n\geq 2} = \frac{\alpha_{n=1}}{T(0)} = \frac{1}{s+f+3}. \tag{3.10}$$

By (3.10), the base penalty of a node is adjusted according to the node's initial trust value such that the base penalty of a node is fixed by the total number of past observations. By this approach, as we will show below, when a node is generating consecutive failures, its trust value will decrease depending on $P[f]$ (= $1-T(0)$). Thus, given the same number of s and f and the same consecutive failures, the total penalty of a node with a higher trust will be less than that of a node with a lower trust.

Theorem 3.2. *The trust function defined above satisfies Property 1 and Property 2.*

Proof. We prove *Property 1* by showing that $PT(n)$ is an increasing function and for $n \geq 2$, $PT(n) - PT(n-1) > PT(n-1) - PT(n-2)$. According to (3.2) and (3.6),

$$PT(n) - PT(n-1) = \alpha \sum_{i=1}^{n} \frac{(1-P[f]^i)}{(1-P[f])}. \tag{3.11}$$

$$PT(n-1) - PT(n-2) = \alpha \sum_{i=1}^{n-1} \frac{(1-P[f]^i)}{(1-P[f])}. \tag{3.12}$$

By (3.11) – (3.12), $(PT(n) - PT(n\text{-}1)) - (PT(n\text{-}1) - PT(n\text{-}2)) = \alpha(1\text{-}P[f]^n)/(1\text{-}P[f]) > 0$, since $\alpha > 0$, $n > 0$ and $0 \leq P[f] < 1$. We assume $P[f] \neq 1$. Thus, $PT(n)$ satisfies *Property 1*.

Next, to prove *Property 2*, let two nodes 1 and 2 have the same number of past behaviors (s_1, f_1) and (s_2, f_2), respectively where $s_1 + f_1 = s_2 + f_2$. Assume that node 1's trust value is higher than node 2's trust value such that $s_1 \geq s_2$ and $f_1 \leq f_2$. They are generating n consecutive failures equally. Let α_1 and α_2 be base penalties of node 1 and node 2, respectively. Then, when $n \geq 2$, by (3.10), $\alpha_1 = \frac{1}{s_1 + f_1 + 3}$ and $\alpha_2 = \frac{1}{s_2 + f_2 + 3}$, and thus $\alpha_1 = \alpha_2$ since $s_1 + f_1 = s_2 + f_2$. Given the same base penalty, since $P[f]$ of node 1 is less than $P[f]$ of node 2 according to (3.2) and (3.5), the total penalty of node 2 is larger than that of node 1. Therefore, the new trust function satisfies *Property 2.* ■

3.3.2.4 State Transition

The hybrid trust model will adaptively choose either the beta trust model or the new trust model based on the occurrence of success or failure. Because the two trust models use different formulas to evaluate trust value on the same input values of s and f (the total number of successes and failures, respectively), they may give different trust values. When we switch from the beta trust model to the new trust model on consecutive failures, we can use the current values of s and f to determine base penalty and then apply the new trust model to evaluate trust value. However, when we switch back to the beta trust model, the current trust value (from the new trust model) will be lower than the value based on the beta trust model (see equation (2.5)). To continue using the beta trust model, we should adjust parameters of the beta trust model such that the beta trust model has the same trust value of our new trust model. We call this procedure *state transition*.

One way to solve the state transition problem is to adjust the number of failures f in the beta trust model. Let T_{New} and T_{Beta} be the trust values in new trust model and the

beta trust model, respectively. By $T_{New} = T_{Beta}$,

$$T_{New} = \frac{s+1}{s+f^*+2}. \tag{3.13}$$

By (3.13), we obtain the adjusted f^* as

$$f^* = \frac{s+1}{T_{New}} - s - 2. \tag{3.14}$$

3.3.2.5 Example of New Trust Model and Consecutive Drops

We demonstrate how our trust model works by using the same example in Section 3.2.2, and compare our trust model with the beta trust model and the entropy trust model. To remind you of the previous example, a node's trust value is approximately 1 after it successfully forwarded 1,000 packets (that is, s = 1,000), then the node starts dropping all received packets. As the number of consecutive drops n goes from 1 to 30, Figure 3.3 shows how their trust values T drop; the trust value in our new trust model is calculated by (3.4) and (3.5). For the first failure (n = 1), our new trust model penalizes the node as the same as the beta trust model does. However, as n grows, our trust model gradually increases the penalty to the node (*Property 1*), and the trust value goes below 0.6 when n = 30. Assuming that the same trust threshold is used, we can see that our trust model detects inside packet drop attackers much faster than the beta trust model and the entropy trust model. In addition, if the node is an intelligent attacker that knows how our trust model works against its packet drops, to avoid being detected by our trust model, the attacker will not launch long number of consecutive drops, which may lead to a failure in dropping consecutive critical packets completely as discussed in Section 3.1. In the rest of paper, we conduct extensive simulations based on the OPNET Modeler to show the

performance of our trust model in practical network environments.

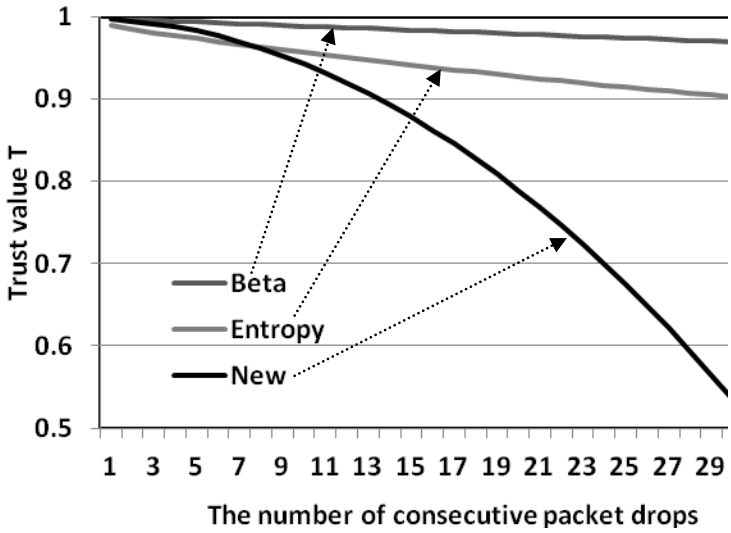


Figure 3.3: New trust model in the presence of consecutive packet drops. After 30 consecutive packet drops, the trust value in the new trust model is very low (i.e., $T_{New} \approx 0.534$) while the trust values in the beta trust model and entropy trust model are still very high (i.e., $T_{Beta} \approx 0.969$ and $T^{*}_{Entr} \approx 0.902$).

3.4 Performance Evaluation

3.4.1 Goals, Metrics, and Methodology

3.4.1.1 Goals of Experiments

Our main goal in this section is to show that our hybrid trust model that considers consecutive drops detects insider packet drop attacks more *quickly* and *accurately* than the beta trust model which does not consider consecutive drops in the packet routing domain. In addition, since a routing algorithm must deliver packets to the destination *reliably* and *energy-efficiently* even under insider packet drop attacks, we show how our hybrid trust model improves both reliability and energy-efficiency of a routing algorithm versus the beta trust model. We discuss reliability and energy-efficiency in more detail in Section 3.4.1.4. We conduct extensive simulations based on the OPNET Modeler.

3.4.1.2 Routing Algorithms for Performance Evaluation

For performance evaluation, we use the following three geographic greedy routing algorithms:

- Geographic greedy routing without a trust model (Pure GRP);
- Pure GRP with the beta trust model (Beta GRP);
- Pure GRP with our hybrid trust model (Hybrid GRP)

The Pure GRP (Geographic Routing Protocol) [21] provided by the OPNET Modeler is a geographic greedy routing that works very similarly with the GPSR that we introduced in Section 2.6. We implemented two trust-based geographic greedy routings

(Beta GRP and Hybrid GRP) based on the GRP and two trust models (the beta trust model and our hybrid trust model) in the OPNET Modeler.

The main difference between the three routings is the neighbor selection algorithm (NSA) by which a sending node S finds a next hop H to forward its packet to H. Algorithm 1 shows the neighbor selection algorithm of the Pure GRP. In the Pure GRP, just like the GPSR, a sending node S selects a next hop H that is the closest one to the destination from its neighbor set **NS**.

Algorithm 1 (NSA in the pure GRP)

Input : Sender S, Destination D, and S's neighbor set **NS**

Output : Next hop node H

1: d_{MIN} := dist (S, D);

2: **for** each node n_i in **NS do**

3: calculate dist (n_i, D);

4: **if** (dist (n_i, D) < d_{MIN}) **then**

5: H := n_i;

6: d_{MIN} := dist (n_i, D);

On the other hand, in the trust-based GRP such as the Beta GRP or the Hybrid GRP, as shown in Algorithm 2, a sending node S finds a next hop H that is the closest one to the destination among its neighbors whose trust value is equal to or higher than the trust threshold *TH*. The only difference between two trust-based GRPs is the difference of trust models that they use.

Algorithm 2 (NSA in the trust-based GRP)

Input : Sender S, Destination D, and S's neighbor set **NS,** Trust threshold *TH*

Output : Next hop node H

1: d_{MIN} := dist (S, D);

2: **for** each node n_i in **NS do**

3: calculate dist (n_i, D) and T(n_i); // T(n_i) is the trust value of node n_i

4: **if** (d (n_i, D) < d_{MIN} **and** T(n_i) ≥ *TH*) **then**

5: H := n_i;

6: d_{MIN} := dist (n_i, D);

3.4.1.3 Packet Drop Attack Models

We use two types of packet drop attacks in simulations.

- Blackhole attack: Drop all received data packets (attack rate is 100%);
- Grayhole attack: Drop received data packets randomly according to various attack rates that range from 20% to 80%.

In our simulations, the attacker drops only data packets, and forwards other types of packets such as control packets. As shown in Figure 3.4, we implemented packet drop attack models at the network layer of a wireless node in the OPNET Modeler, since a node can determine whether a received packet is either a data packet or a control packet when the packet reaches the network layer. When an attacker receives a data packet from a node, the attacker sends an ACK message back to the sender, and then decides whether it drops the data packet based on a pre-defined attack rate. For example, if the attack rate is 100%, the (blackhole) attacker drops all received packets. If the attack rate is 50%, the

(grayhole) attacker randomly drops 50% of the received data packets. If an attacker decides to forward a data packet, it simply transmits the packet to a next hop node (a relay node or the BS).

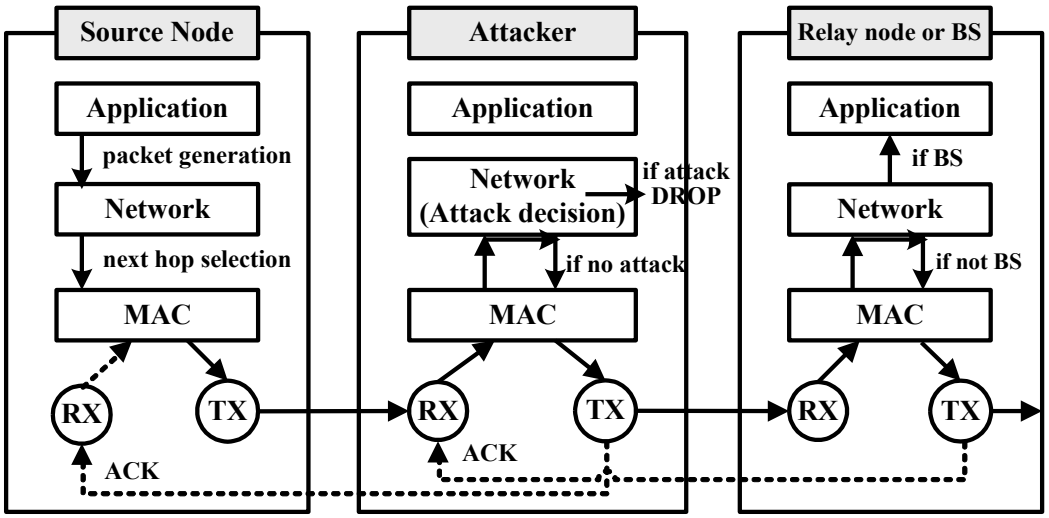


Figure 3.4: Implementation of packet drop attacks models in the OPNET Modeler. When an attacker receives a data packet from a node, the attacker sends an ACK message back to the sender, and then decides whether it drops the data packet based on a pre-defined attack rate.

3.4.1.4 Performance Evaluation Metrics

We compare the Pure GRP, the Beta GRP, and our proposed Hybrid GRP in terms of the following performance evaluation metrics:

1) Detection completion time (DCT): *DCT* is defined as the simulation time (seconds) when all attackers are detected by victim nodes whose packets are dropped by the attackers. *DCT* evaluates how quickly a trust model detects attackers.

2) Detection rate (DR) and False alarm rate (FAR): As shown in Table 3.2, node *i* classifies node *j* into either *trustworthy* or *untrustworthy*, based on node *j*'s trust value at

node *i*. Node *i* may either correctly detect an attacker as an untrustworthy node (*Hit*) or fail in detecting it (*Miss*). *DR* is the probability that an attacker is being considered as an untrustworthy node. On the other hand, a trust model may misclassify a normal node (non-attacker) into an untrustworthy node (*False alarm*). We get *FAR* as the number of false alarms divided by the number of normal nodes as defined in [23]. Unlike *DR* ($0 \leq DR \leq 1$), this *FAR* can be over 1.0, since a node can be falsely detected multiple times by its neighbors. We use *DR* and *FAR* to evaluate the detection accuracy of a trust model.

Table 3.2: Detection and false alarm in trust-based attacker detection problem

	Decision result	
Given node is	Untrustworthy	Trustworthy
Attacker	Hit (Detection)	Miss
Non-attacker	False alarm	Correct rejection

3) Packet delivery rate (PDR): *PDR* is the probability that a data packet is successfully delivered to the base station. A reliable trust-based routing will deliver most of the data packets to the destination even in the presence of inside packet drop attackers in the network. We use *PDR* to evaluate the *reliability* of a trust-based routing.

4) Total energy consumption (E_T) and Energy efficiency (e_S): For energy consumption analysis, we consider the total amount of energy consumed by transmitting and receiving data packets, E_T (*J*). To get E_T, we use the first order radio model [26]. In this model, the amounts of energy required to send a *k* bit packet over distance *d*, E_{Tx}, and to receive a *k* bit packet, E_{Rx}, are

$$E_{Tx}(k,d) = E_{elec} \times k + E_{amp} \times k \times d^2 \tag{3.15}$$

$$E_{Rx}(k,d) = E_{elec} \times k \tag{3.16}$$

where E_{elec} is the energy dissipation of the radio to run the transmitter and receiver circuitry and is $50nJ/bit$, and E_{amp} is the energy dissipation for the transmit amplifier and is $100pJ/bit/m^2$. By using (3.15) and (3.16), we get the total energy consumption for data packet delivery E_T by summing E_{Tx} and E_{Rx} for all nodes in the network. In addition, to see how efficiently a routing works in terms of energy, we use the energy efficiency e_S (mJ) that is defined as

$$e_S = \frac{E_T}{N_S} \tag{3.17}$$

where N_S is the total number of packets successfully delivered to the base station. The energy efficiency e_S indicates the average energy consumption to successfully deliver one data packet to the destination.

5) Other metrics (N_T, N_A, and N_{NA}): N_T is the total number of data packets generated by all source nodes, N_A is the total number of data packets dropped due to attacks, and N_{NA} is the total number of packets dropped due to other network problems (non-attacks) such as collision or noise.

3.4.1.5 OPNET Simulation Setup

The parameters in Table 3.3 are used in our simulations. An ad hoc WSN with 50 sensor nodes is built in a 1km × 1km area as shown in Figure 3.5. We place a single base station (BS) at one corner and every other sensor will generate and send data packets to the BS. We use the default OPNET setting of data packet size (1,024 bits) and

transmission power to 0.001W. For the channel access model, we use CSMA/CA (Carrier Sense Multiple Access/Collision Avoidance) with maximum seven retransmissions (default value in OPNET). Each node generates a data packet randomly within each 10-second period since the simulation time 100 seconds elapse, and forwards it toward the BS based on its routing algorithm such as the Pure GRP, the Beta GRP, and our Hybrid GRP. For trust models, we use three different trust thresholds (*TH*s) of 0.6, 0.7, and 0.8. Considering a practical assumption that attackers develop a high trust value before launching attacks [20], we set an initial trust value of each node to be approximately 0.99. We place single or multiple attackers near the BS so that attackers can receive many data packets from nodes. Attackers start launching pre-defined attacks (blackhole attack or grayhole attack) once they receive data packets. By considering our simulation PC's performance (Pentium CPU 3GHz and RAM 2GB), we set the maximum simulation time to be three hours (10,800 seconds) to find Detection Completion Time *DCT*.

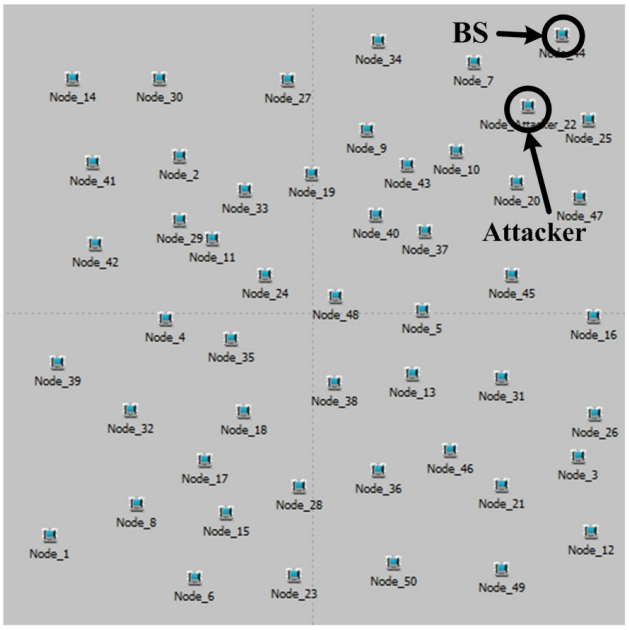


Figure 3.5: A random WSN topology in a 1km × 1km area.

Table 3.3: OPNET simulation setup parameters

Parameters		Setting
General	Terrain dimension	1km × 1km
	Number of nodes	50
	Topology	Random
	Max. simulation time	3 hours (10,800 seconds)
	Number of runs	10
	Base routing algorithm	GRP
Sensor	Transmission power	0.001W (OPNET default)
	Max. retransmissions	7 (OPNET default)
Data packet Generation	Start time – Stop time	100 seconds – end of simulation
	Destination	Base station
	Packet arrival interval	Every 10 second
	Packet size	1,024 bits
Trust model	Type	None, Beta, and Hybrid
	Initial trust value	0.99
	Trust threshold (*TH*)	0.6, 0.7, and 0.8
Attack model	Number of attackers	Single attacker and Multiple attackers (3, 5, 8)
	Attack type	Blackhole (attack rate 100%) and Grayhole (attack rate 20% - 80%)

3.4.2 Simulation Results

We first compare three routing algorithms in the presence of the single attacker and multiple attackers in a WSN without bursty transmission errors. Then, we compare them in a WSN with bursty transmission errors.

3.4.2.1 Single Attacker

Table 3.4 shows all simulation results in a single attacker case. All performance evaluation metrics in the first column are introduced in Section 3.4.1.4. When DCTs of both trust-based routings (Beta GRP and our Hybrid GRP) are found within maximum simulation time (three hours), for comparison, we show the value of each metric measured at the slower one's *DCT*. When any of trust-based routing fails in finding *DCT* within the maximum simulation time, we show all metrics measured at the maximum simulation time (three hours). The Pure GRP is compared to see how seriously the attacker can degrade the network performance when we do not use any trust model.

Table 3.4: Simulation results (Single Attacker): *DCT*: Detection completion time; *FAR*: False alarm rate; N_A: the total number of data packets dropped due to attacks; N_{NA}: the total number of packets dropped due to other network problems (non-attacks) such as collision or noise; *PDR*: Packet delivery rate; E_T: Total energy consumption; e_S: Energy efficiency.

(a) $TH = 0.6$

Attack rate %		100	80	60	40	30	20
DCT	Beta	781	1155	2095	*0.04	*0.02	*0.02
	Hybrid	183	244	457	1846	*0.88	*0.02
FAR	Beta	0	0.037	0.039	0.097	0.104	0.097
	Hybrid	0.072	0.102	0.118	0.127	0.125	0.112
N_A	GRP	661	820	1115	3883	2914	1894
	Beta	331	397	551	4176	3177	2078
	Hybrid	44	64	96	265	989	2139
N_{NA}	GRP	336	509	946	4942	4931	4964
	Beta	364	515	858	2690	2702	2703
	Hybrid	282	364	591	1983	2002	1859
PDR	GRP	0.711	0.747	0.790	0.828	0.847	0.866
	Beta	0.799	0.825	0.856	0.866	0.885	0.906
	Hybrid	0.906	0.918	0.929	0.956	0.941	0.922
E_T (J)	GRP	84.9	129.8	242.1	1272.1	1272.9	1274.8
	Beta	86.0	131.7	245.6	1290.2	1290.5	1292.9
	Hybrid	88.0	133.9	248.7	1305.7	1306.0	1293.3
e_S (mJ)	GRP	34.61	32.98	31.34	29.88	26.97	28.63
	Beta	31.14	30.27	29.35	28.97	27.50	27.73
	Hybrid	28.00	27.67	27.35	26.57	26.57	27.29

(b) $TH = 0.7$

Attack rate %		100	80	60	40	30	20
DCT	Beta	522	774	1083	4190	*0.16	*0.02
	Hybrid	151	241	396	1123	2030	*0.02
FAR	Beta	0.002	0.037	0.045	0.125	0.114	0.112
	Hybrid	0.081	0.108	0.116	0.141	0.135	0.131
N_A	GRP	427	534	595	1643	2914	1894
	Beta	213	236	278	580	2858	2113
	Hybrid	39	53	77	163	255	2134
N_{NA}	GRP	225	335	479	1975	4931	4964
	Beta	243	336	460	1190	2320	2118
	Hybrid	188	251	304	828	1853	1719
PDR	GRP	0.712	0.747	0.784	0.828	0.847	0.866
	Beta	0.798	0.833	0.850	0.904	0.899	0.917
	Hybrid	0.899	0.911	0.922	0.947	0.958	0.925
E_T (J)	GRP	55.2	84.5	121.3	498.8	1272.9	1274.8
	Beta	56.0	86.0	123.2	511.7	1295.1	1291.7
	Hybrid	57.0	87.1	124.6	512.5	1304.7	1293.6
e_S (mJ)	GRP	34.29	32.95	31.48	29.87	29.26	28.63
	Beta	31.18	30.06	29.50	28.05	28.00	27.41
	Hybrid	28.06	27.86	27.48	26.80	26.49	27.19

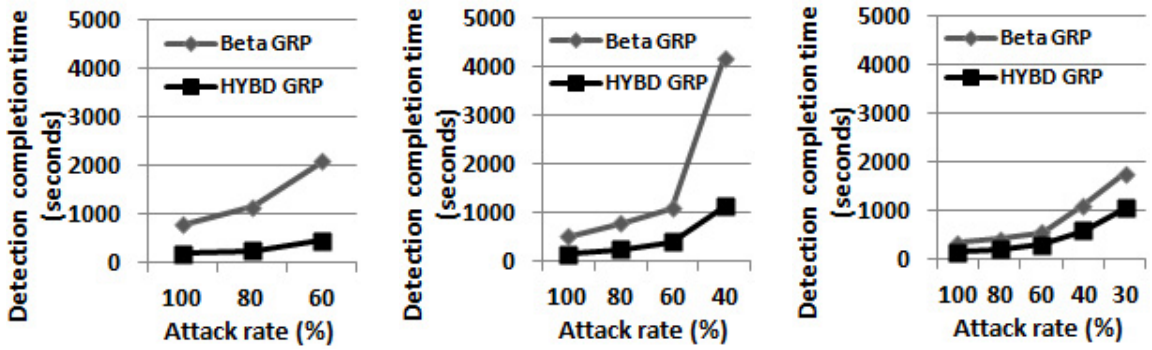
(c) $TH = 0.8$

Attack rate %		100	80	60	40	30	20
DCT	Beta	342	424	558	1112	1771	*0.12
	Hybrid	151	212	295	597	1062	7376
FAR	Beta	0.014	0.039	0.043	0.104	0.118	0.141
	Hybrid	0.072	0.087	0.110	0.133	0.141	0.189
N_A	GRP	256	280	288	397	474	1894
	Beta	123	128	139	183	240	2063
	Hybrid	29	42	57	84	120	432
N_{NA}	GRP	141	182	243	508	838	4964
	Beta	148	183	231	375	482	1759
	Hybrid	121	142	175	259	365	1474
PDR	GRP	0.708	0.739	0.779	0.822	0.843	0.866
	Beta	0.801	0.823	0.845	0.888	0.912	0.925
	Hybrid	0.890	0.894	0.903	0.932	0.941	0.962
E_T (J)	GRP	33.1	43.3	59.1	125.1	203.8	1274.8
	Beta	33.7	43.8	60.0	127.8	208.4	1294.2
	Hybrid	34.3	44.2	60.7	128.4	208.9	1304.9
e_S (mJ)	GRP	34.63	33.31	31.66	30.06	29.35	28.63
	Beta	31.08	30.27	29.65	28.40	27.74	27.18
	Hybrid	28.28	28.17	27.97	27.20	26.95	26.35

We explain the results in detail in terms of detection speed, detection accuracy, routing reliability, and energy-efficiency in turn.

1) Detection speed and the damage to the network

To compare how *quickly* each trust model detects the inside packet drop attacker, we first examine the *DCT* of each routing algorithm as shown in Table 3.4 and Figure 3.6 When *DCT* is not found within three hours, we show *DR* instead; we will explain *DR* in detail in *2) Detection accuracy*. In general, when *TH* is fixed, as the attack rate grows, *DCT* decreases because the attacker drops more packets, its trust value decreases faster, and thus it is detected quicker by trust models. When the attack rate is fixed, as *TH* grows, *DCT* decreases because a trust model with a higher *TH* detects the attacker faster than the same trust model with a lower *TH*.



(a) *TH* = 0.6 (b) *TH* = 0.7 (c) *TH* = 0.8

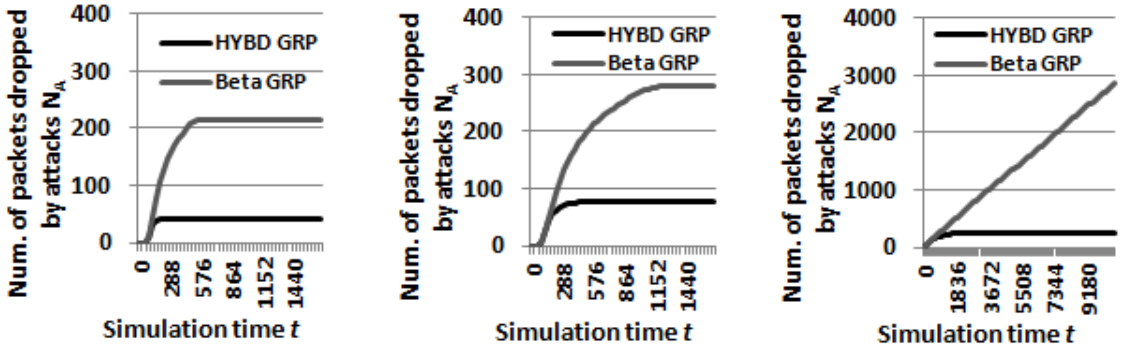
Figure 3.6: Detection completion time *DCT* (Single attacker). Our Hybrid GRP detects an inside packet drop attacker always faster than the Beta GRP in the presence of various attack models.

The results show the following:

First, our Hybrid GRP can detect an inside packet drop attacker much faster than the Beta GRP under various attack models, regardless of *TH*. When both trust models detect the attacker, our hybrid trust model can detect it at least 2.7 times and at most 4.7

times faster than the beta trust model. We found that when the attack rate is set to be 100 × (1−*TH*), the beta trust model fails in detecting the attacker (i.e., *DCT* was not found) but our trust model can detect it. However, when the attack rate is set to be too low, both trust models failed in detecting the attacker.

Second, detection speed is closely related to the damage to the network caused by the attackers. The early detection capability of our hybrid trust model based on consecutive drops significantly reduces the damage to the network due to attacks compared to that of the beta trust model. In case of the Pure GRP, N_A continuously grows as the simulation proceeds since it does not have a trust mechanism, and thus cannot defend against the attacker. For the Beta GRP and our Hybrid GRP, when *DCT* is found, N_A stops growing because the attacker is completely detected by all victim nodes (see Figure 3.7(a) and Figure 3.7(b). In addition, we can see the Beta GRP's N_A is more than 3.5 times larger than our Hybrid GRP's N_A. Also, as shown in Figure 3.7(c), when the Beta GRP cannot find the attacker, N_A grows continuously, and this degrades the performance of a routing as we will discuss in detail in *3) Reliability*.



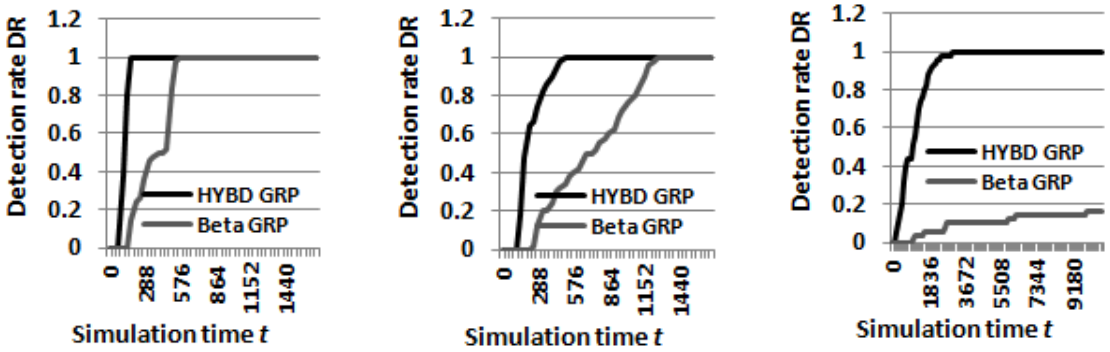
(a) Attack rate = 100% (b) Attack rate = 60% (c) Attack rate = 30%

Figure 3.7: The total number of packets dropped due to attacks N_A when *TH* = 0.7 (Single attacker). The early detection capability of our hybrid trust model based on consecutive drops significantly reduces the damage to the network due to attacks compared to that of the beta trust model.

2) Detection accuracy

In addition to early detection capability, a good trust model must have high detection accuracy, which can be evaluated by using *DR* and *FAR*. Thus, a desired trust model has a high *DR* and a low *FAR*. A comparison of the detection accuracy of the beta trust model and our hybrid trust model is as follows:

First, our hybrid trust model can detect an inside packet drop attacker not only more quickly but also more accurately than the beta trust model. We already demonstrated that our hybrid trust model can detect the attacker much faster than the beta trust model in *1) Detection speed and the damage to the network.* Figure 3.8 shows how *DR*s of the Beta GRP and our Hybrid GRP change as the simulation time *t* proceeds when *TH* is 0.7. In this figure, the value of (1 – *DR*) at time *t* indicates the percentage of victim nodes that cannot detect the attacker. That is, the difference between *DR*s of two trust models shows the difference of attacker detection accuracy of two trust models. The *DR* of our hybrid trust model is always above that of the beta trust model, which means that victim nodes with our hybrid trust model always detect the attacker faster and more accurately than victim nodes with the beta trust model.



(a) Attack rate = 100% (b) Attack rate = 60% (c) Attack rate = 30%

Figure 3.8: Detection rate *DR* when *TH* = 0.7 (Single attacker). The difference of *DR*s shows the difference of attack detection accuracy of two trust models (Beta and hybrid trust models).

In particular, the beta trust model cannot completely detect the attacker when the attack rate is set to be equal to or less than $100 \times (1 - TH)$. Meanwhile, our hybrid trust model can detect the attacker whose attack rate is $100 \times (1 - TH)$. For example, in Figure 3.8(c), when the attack rate = 30% and TH = 0.7, more than 80% of victim nodes with the beta trust model fail to detect the attacker within the maximum simulation time (10,800 seconds). Meanwhile, all victim nodes with our hybrid trust model detected the attacker in the early stage (within 2,030 seconds). This is due to the limitation of the design of the beta trust model. That is, in the beta trust model, if an attacker continues to drop 30% of received packets randomly, its trust value T gradually lowers to 0.7 according to (2.5). In practice, the attacker is likely to start dropping packets after it has a high trust value [20] to avoid being detected by a trust model. For this reason, such an attacker's trust value hardly falls below 0.7, and thus the beta trust model with $TH = 0.7$ fails to detect it. On the other hand, our hybrid trust model significantly penalizes such an attacker's repeating consecutive drops, and thus can detect the attacker with the same trust threshold. However, when the attack rate is less than $100 \times (1 - TH)$, both the beta trust model and our hybrid trust model failed to detect the attacker. When the attacker knows the trust threshold, it can estimate its trust value before dropping any packet to make sure that he/she will never be caught. Although our hybrid trust model cannot defend against such an attacker, the damage that the attacker can cause to the network is limited according to TH. For example, when TH is 0.7, the attacker cannot drop more than 30% of packets without being caught. It will be interesting (and challenging) to study this problem, for example from the point of view of Game Theory [13]. However, since a detailed discussion to address this problem is out of the scope of this dissertation, we leave it to

our future work.

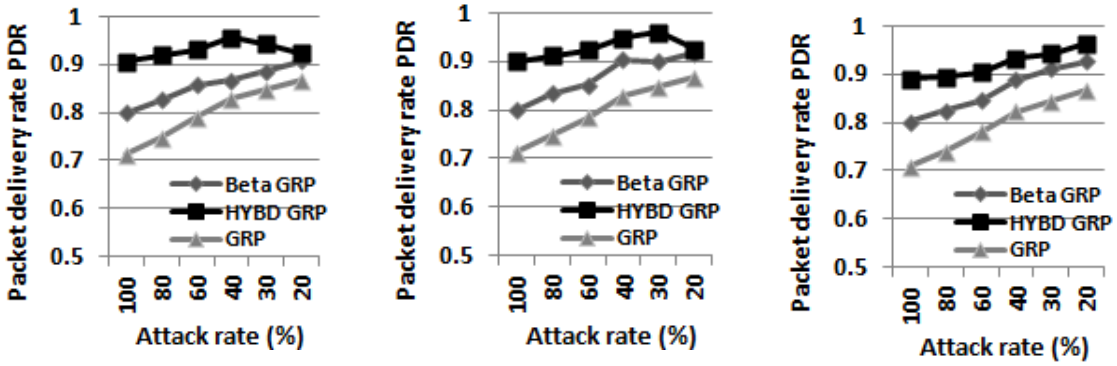
Second, our hybrid trust model generates a small *FAR* that is slightly higher than that of the beta trust model. High *FAR* must be avoided, because it may negatively affect network connectivity, decrease the number of available relay nodes in the network, and thus significantly degrade the packet delivery performance of a routing protocol. As shown in Table 3.4, we see that both our Hybrid GRP and Beta GRP do not have a high *FAR*. Our Hybrid GRP has a slightly higher *FAR* than that of the Beta GRP since our hybrid trust model gives a larger penalty to a node generating consecutive drops than the beta trust model. In practice, packets can be dropped consecutively due to some network problems as well as attacks. Therefore, this result was expected. In fact, as we will discuss below, this small *FAR* does not degrade the packet delivery performance of our Hybrid GRP but helps to improve its the packet delivery performance. Based on this result, we consider that this small FAR is acceptable in order to get high detection accuracy.

3) Routing reliability

The final goal of a routing algorithm in a WSN is to deliver data packets from source nodes to the BS. Therefore, a good trust-based routing algorithm should be able to deliver packets to the destination successfully even under insider packet drop attacks. We examine the routing reliability of each routing based on *PDR*. Recall that a single attacker is deployed near the BS to receive the significant number of data packets from many nodes throughout the network.

As shown in Table 3.4, our Hybrid GRP has the highest *PDR* among three routing algorithms and can deliver at least more than 89% of packets to the BS in all simulation setups. This result shows that our Hybrid GRP can route packets very reliably even under

various packet drop attacks. Specifically, our Hybrid GRP improves *PDR* of the Pure GRP by at most 19% and PDR of the Beta GRP by at most 10% (when the attack rate =100% and *TH*=0.6 as shown in Figure 3.9(a)).



(a) *TH* = 0.6 (b) *TH* = 0.7 (c) *TH* = 0.8

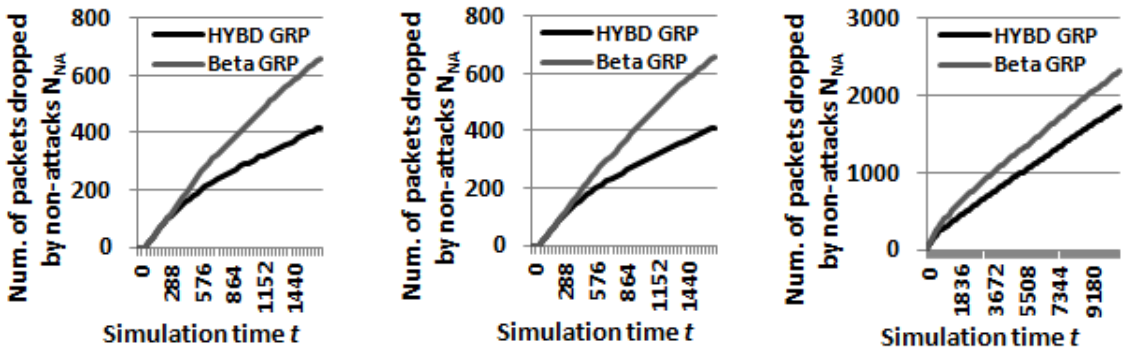
Figure 3.9: Packet delivery rate *PDR* (Single attacker). Our Hybrid GRP has the highest *PDR* among three routing algorithms (GRP, Beta GRP, and Hybrid GRP) and can deliver at least more than 89% of packets to the BS in all simulation setups.

Since *PDR* is determined by two metrics N_A and N_{NA}, we examine them in turn.

First, as we explained in *1) Detection speed*, our Hybrid GRP has the smallest N_A among three routing algorithms as shown in Figure 3.7, because our Hybrid GRP detects the attacker quickly based on consecutive drops and finds a new path to the BS through a trustworthy node. The Beta GRP has larger N_A than our Hybrid GRP because it has slower detection speed than our Hybrid GRP. Since the Pure GRP does not use any trust model, it has the most significant amount of packet loss by the attacker.

Second, our Hybrid GRP has the least N_{NA} among three routing algorithms. Packets can be naturally dropped in a WSN because of many reasons such as collision

and congestion. The OPNET simulates channel quality problems such as collision and congestion. As shown in Table 3.4, our Hybrid GRP and the Beta GRP has less N_{NA} than that of the Pure GRP. This is because a trust-based routing algorithm can avoid bad communication links by classifying normal nodes (non-attackers) with many transmission errors as untrustworthy nodes. This causes both the Beta GRP and our Hybrid GRP to generate a small *FAR*, as we discussed before, but it improves the *PDR* of a trust-based routing algorithm. For the same reason as discussed in *1) Detection speed*, when such normal nodes in bad links generate many consecutive drops, our Hybrid GRP penalizes them based on the number of consecutive drops and thus quickly avoids them. As a result, our Hybrid GRP has less N_{NA} than the Beta GRP as shown in Figure 3.10. In practice, it is often assumed that WSNs have hundreds of sensors in order to be tolerant to many faulty sensors and thus a small *FAR* may not be a problem.



(a) Attack rate = 100% (b) Attack rate = 60% (c) Attack rate = 30%

Figure 3.10: The total number of packets dropped due to non-attacks N_{NA} when TH = 0.7 (Single attacker). Our Hybrid GRP has the least N_{NA} among three routing algorithms (GRP, Beta GRP, and Hybrid GRP) by penalizing nodes in bad links based on the number of consecutive drops and thus quickly avoids such nodes.

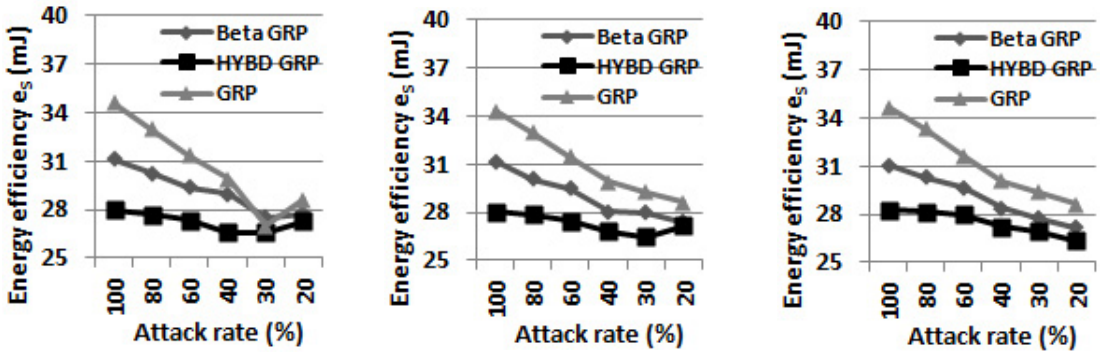
4) Energy-efficiency

We now compare the energy efficiency of each routing algorithm. The results show the following:

First, the total energy consumption of each routing algorithm is as follows: E_T of our Hybrid GRP > E_T of the Beta GRP > E_T of the Pure GRP as shown in Table 3.4. E_T is closely related to *PDR*. In the Pure GRP, a sender continues to forward data packets to a neighbor closest to the BS even though an attacker keeps dropping the packets. This is because the Pure GRP does not have a mechanism to monitor neighbors' packet transmission behaviors. The Pure GRP has the least *PDR*, which means the largest amount of data packets are dropped before reaching to the BS. Consequently, the Pure GRP has the least amount of energy consumption for packet transmissions and receptions. Likewise, the Beta GRP has less E_T than our Hybrid GRP. In addition, a trust-based routing like our Hybrid GRP and the Beta GRP are likely to have longer average hops or longer average routing distance from a source to the BS than the Pure GRP. This is because a trust-based routing avoids an inside packet drop attacker and normal nodes with many transmission errors, and then finds another node. The avoided nodes are originally chosen optimal nodes in terms of minimum hops; we note that the greedy geographic routing approximates the minimum hop routing [8], and thus we assume that the avoided node is likely to be in the shortest path to the BS. Since E_T does not correctly evaluate the energy-efficiency of a routing, we use the energy-efficiency e_S for energy analysis.

Second, our Hybrid GRP consumes the least energy to deliver a packet to the BS among three routing algorithms. As shown in Table 3.4, the energy-efficiency e_S of each

routing is as follows: e_S of Pure GRP > e_S of Beta GRP > e_S of Hybird GRP. Especially, as shown in Figure 3.11, when the attack rate = 100% and TH = 0.6, our Hybrid GRP can save approximately 19% of energy as compared to the Pure GRP and 10% of energy as compared to the Beta GRP, which correspond to 6.61 *mJ* and 3.14 *mJ*, respectively. To help you understand, according to the first radio model (3.15) and (3.16), 6.61 *mJ* is the amount of energy that a node consumes to transmit a 1,024-bit packet to another node that is approximately 253m away.



(a) TH = 0.6 (b) TH = 0.7 (c) TH = 0.8

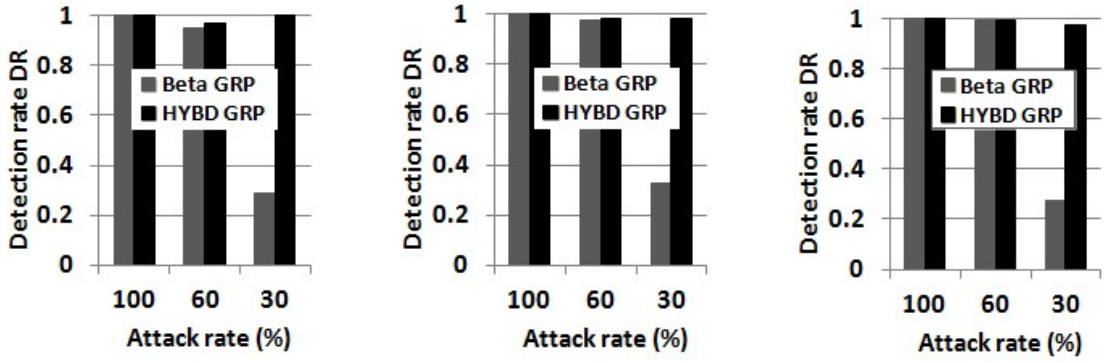
Figure 3.11: Energy efficiency e_S. (Single attacker). Our Hybrid GRP consumes the least energy to deliver a packet to the BS among three routing algorithms (GRP, Beta GRP, and Hybrid GRP).

3.4.2.2 Multiple Attackers

We examine how each routing algorithm performs in the presence of multiple attackers. We assume that attackers are not colluding. To deploy multiple attackers in the network, we randomly choose 5, 8, and 10 relay nodes as attackers, which correspond to 5%, 10%, and 15% of total nodes in the network, respectively. We use three attack rates such as 30%, 60%, and 100%. For the trust threshold, we use 0.7. We compare all performance evaluation metrics recorded at the maximum simulation time (three hours). Since we thoroughly examined the detection speed of each routing in the single attacker

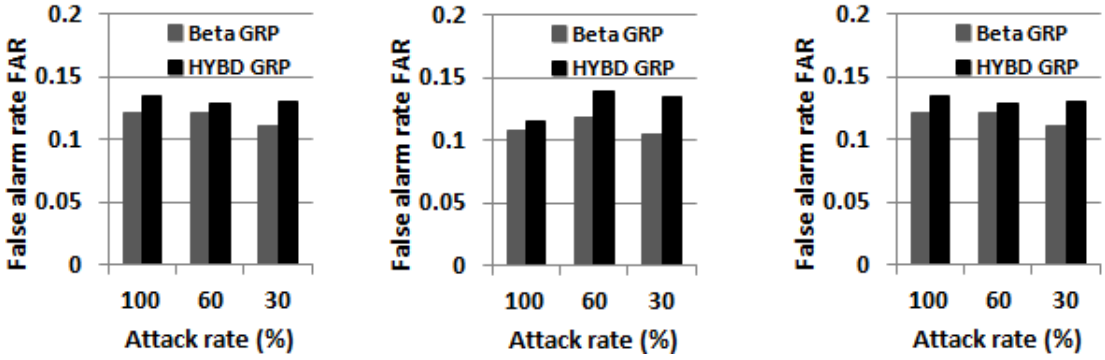
case, we do not provide *DCT* here. In summary, the results in the multiple attacker case are similar with the results in the single attacker case. We briefly report the results.

First, we report *DR* and *FAR*. Like the single attacker case, as shown in Figure 3.12, our hybrid trust model based on consecutive drops has a very high *DR* in all simulations setups. Meanwhile, the Beta GRP has a low *DR* in the presence of multiple grayhole attackers with 30% of attack rate. On the other hand, our hybrid trust model also has a slightly higher *FAR* than the beta trust model (see Figure 3.13). However, as we will show *PDR* below, this small *FAR* (due to nodes in bad links) does not degrade the packet delivery performance of our Hybrid GRP but improves it.



(a) 3 attackers (5%) (b) 5 attackers (10%) (c) 8 attackers (15%)

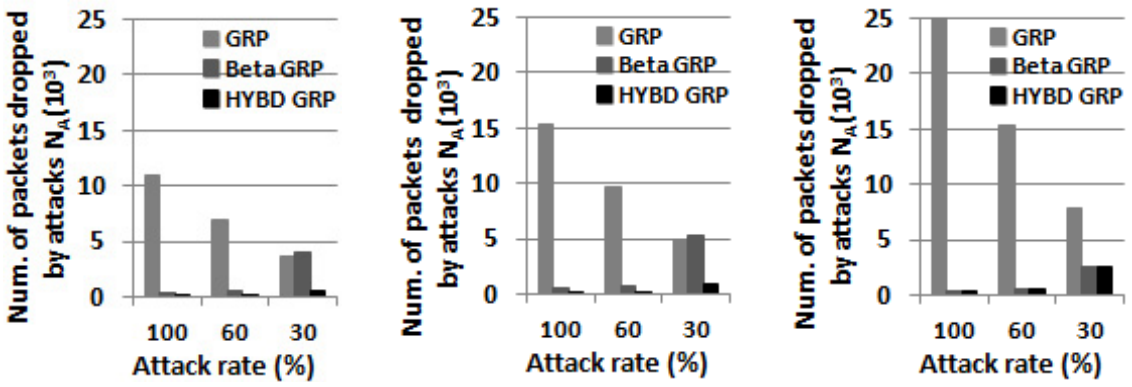
Figure 3.12: Detection rate *DR* (Multiple attackers). Our hybrid trust model based on consecutive drops has a very high *DR* in all simulations setups.



(a) 3 attackers (5%) (b) 5 attackers (10%) (c) 8 attackers (15%)

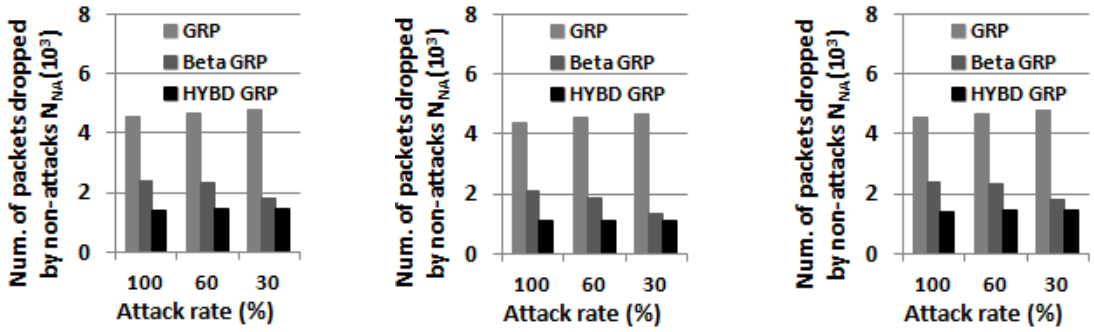
Figure 3.13: False alarm rate *FAR* (Multiple attacks). Our hybrid trust model has a slightly higher *FAR* than the beta trust model.

Second, we report N_A, N_{NA}, and *PDR*. In summary, as shown in Figure 3.14, Figure 3.15, and Figure 3.16, our Hybrid GRP has the least N_A and N_{NA}, and the highest *PDR* among the three routing algorithms in the presence of multiple attackers. As shown in Figure 3.14, it is not surprising that N_A grows as the number of attackers grows. Especially, in case of the Pure GRP, we can see that multiple attackers (especially when they are blackhole attackers) drop a very large number of packets since the Pure GRP does not have any defensive mechanism such as the trust mechanism against the multiple attackers. Meanwhile, both our Hybrid GRP and the Beta GRP have much less N_A than the Pure GRP since their trust mechanisms can defend against the multiple attackers. In addition, thanks to the trust evaluation scheme using consecutive drops, our Hybrid GRP is able to further decrease N_A compared with the Beta GRP. On the other hand, N_{NA} does not significantly change as the number of attackers grows as shown in Figure 3.15. As in the single attacker case, our Hybrid GRP has the least N_{NA} because some normal nodes (non-attackers) with many transmission errors are detected (as false alarms) by our hybrid trust model. Last, our Hybrid GRP has the highest *PDR* as shown in Figure 3.16. We clearly see that our Hybrid GRP can deliver most of packets to the BS very reliably even in the presence of multiple attackers with various attack rates. It is particularly impressive that our Hybrid GRP can deliver approximately 92% of data packets from source nodes to the BS successfully even when 15% of nodes in the network are grayhole attackers that are randomly dropping 30% of packets while the Pure GRP and the Beta GRP can deliver only 71% and 78% of data packets to the BS, respectively. In other words, when compared with the Pure GRP, the Beta GRP without considering consecutive drops can only improve *PDR* by 7%, but our Hybrid GRP can significantly improve *PDR* by 21%.



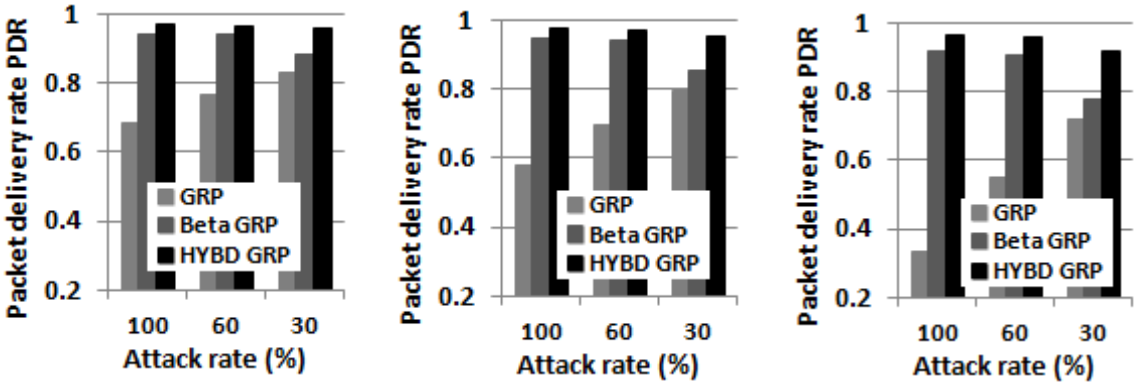
(a) 3 attackers (5%) (b) 5 attackers (10%) (c) 8 attackers (15%)

Figure 3.14: The number of packet drops due to attacks N_A (Multiple attackers). Our Hybrid GRP has the least N_A among three routing algorithms (GRP, Beta GRP, and Hybrid GRP).



(a) 3 attackers (5%) (b) 5 attackers (10%) (c) 8 attackers (15%)

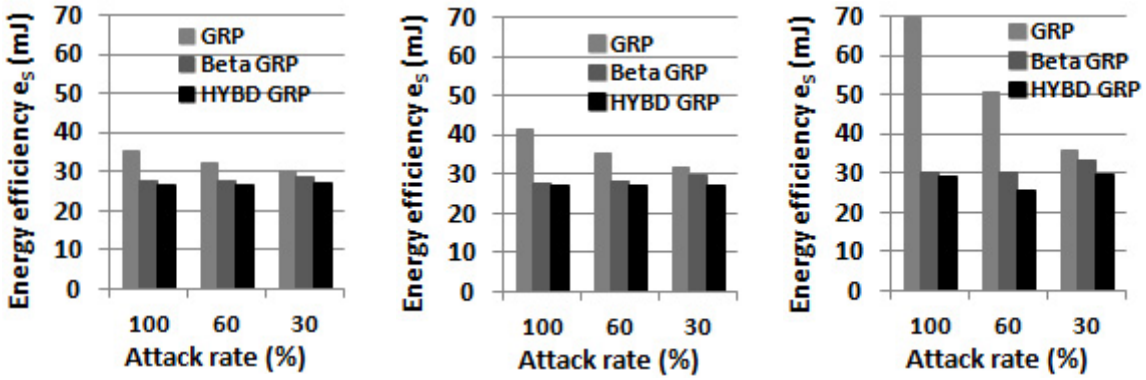
Figure 3.15: The number of packet drops due to non-attacks N_{NA} (Multiple attackers). Our Hybrid GRP has the least N_{NA} among three routing algorithms (GRP, Beta GRP, and Hybrid GRP).



(a) 3 attackers (5%) (b) 5 attackers (10%) (c) 8 attackers (15%)

Figure 3.16: Packet delivery rate *PDR* (Multiple attackers). Our Hybrid GRP has the highest *PDR* among three routing algorithms (GRP, Beta GRP, and Hybrid GRP).

Finally, we report e_S. Our Hybrid GRP has the least e_S among the three routing algorithms as shown in Figure 3.17. Like the single attacker case, our Hybrid GRP consumes the smallest amount of energy to deliver a data packet to the BS. In practice, the network operation time is much greater than our maximum simulation time (three hours). Therefore, when there are inside packet drop attackers in the network, the Pure GRP and the Beta GRP will waste a lot of energy for unsuccessful packet transmissions.



(a) 3 attackers (5%) (b) 5 attackers (10%) (c) 8 attackers (15%)

Figure 3.17: Energy efficiency e_S (Multiple attackers). Our Hybrid GRP has the least e_S among three routing algorithms (GRP, Beta GRP, and Hybrid GRP).

3.4.2.3 Performance Comparison in a WSN with Temporal Bursty Errors

In this section, we examine how our hybrid trust model works in a WSN where packets can be dropped consecutively due to fading or temporal obstacles as well as insider packet drop attacks.

We simulate such WSN in OPNET as follows. We deploy two blackhole attackers near the BS. In our simulation setup, 4% of packets are dropped naturally. We consider this natural packet drop rate as a normal error rate when there are no temporal errors in a WSN. Let the temporal error factor k be 1 at the normal error rate. To simulate temporal

bursty errors in a WSN, we temporarily increase the error rate by multiplying the normal error rate (=4%) by various k (up to 25) during some time duration (say, temporal error period). Then, during the period, packets are dropped with the increased error rate. We randomly choose 40% of nodes in which packets are dropped with various error rates during various error periods such as 600s, 1,200s, and 1,800s. The temporal error period starts randomly during the simulation. The higher error rate and the longer error period, the more packets are dropped during the error period; thus, the higher chance that packets are dropped consecutively.

We explain results by using two performance metrics such as false alarm rate and packet drop rate.

First, our hybrid trust model does not generate a high false alarm even in a WSN with various error rates and error periods. As shown in Table 3.5, in most setups except when the error rate is extremely high, the false alarm rate of our hybrid trust model is the same as that in a normal error rate. This is because the large consecutive drops are not likely to occur even when the temporal error rate is higher than the normal error rate. Meanwhile, when the error rate is set to be extremely high such as 100%, the false alarm rate of our hybrid trust model is slightly increased to 0.11 because, in this setup, all packets are dropped consecutively, but the false alarm rate of the beta trust model is also increased to 0.02 when the temporal error period is 1,800 seconds.

Second, our hybrid trust model significantly reduces the packet drop rate of a routing even in a WSN with various error rates and error periods. As shown in Table 3.5, when either the error rate or the error period grows, the packet drop rate of each routing also grows because packets are dropped more during the temporal error period. Figure

3.18 shows in detail the reasons for packet drops at various error rates when the error period is set to be 1,200 seconds; in the figure, the number of packet drops due to attacks, temporal errors and other network problems are depicted as Attack (black), Temp (gray) and Others (white), respectively. When the Pure GRP is used, most of packet drops occur due to the two undetected blackhole attackers. Meanwhile, since both trust-based routings can catch the attackers, they can dramatically reduce the packet drop rate compared to the Pure GRP. In fact, our hybrid trust model avoids problematic nodes under the influence of temporal errors or collisions as well as attackers much faster than the beta trust model. Consequently, our hybrid trust model can further reduce packet drops.

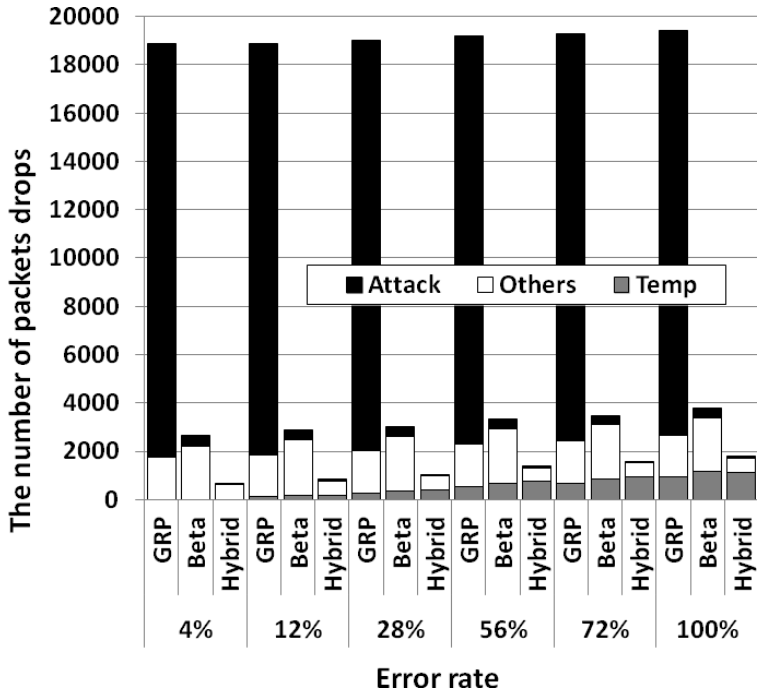


Figure 3.18: The number of packet drops at various error rates when the error period is set to be 1,200 seconds. Our hybrid trust model significantly reduces the packet drop rate of a routing even in a WSN with various error rates and error periods

Table 3.5: False alarm rate and packet drop rate in a WSN with temporal bursty errors

(a) Error period = 600 seconds

Error period		600s				
Error rate %	*k*	FAR		Packet Drop Rate %		
		Beta	Hybrid	GRP	Beta	Hybrid
4	1	0	0.09	35.96	5.03	1.30
12	3	0	0.09	36.07	5.29	1.46
28	7	0	0.09	36.20	5.58	1.65
56	14	0	0.09	36.34	5.82	1.84
72	18	0	0.09	36.40	5.92	2.25
100	25	0	0.11	36.56	6.40	2.50

(b) Error period = 1,200 seconds

Error period		1,200s				
Error rate %	*k*	FAR		Packet Drop Rate %		
		Beta	Hybrid	GRP	Beta	Hybrid
4	1	0	0.09	35.96	5.03	1.30
12	3	0	0.09	36.03	5.45	1.64
28	7	0	0.09	36.27	5.73	1.98
56	14	0	0.09	36.61	6.33	2.65
72	18	0	0.09	36.75	6.62	3.02
100	25	0	0.11	37.02	7.22	3.39

(c) Error period = 1,200 seconds

Error period		1,200s				
Error rate %	*k*	FAR		Packet Drop Rate %		
		Beta	Hybrid	GRP	Beta	Hybrid
4	1	0	0.09	35.96	5.03	1.30
12	3	0	0.09	36.17	5.53	1.66
28	7	0	0.09	36.36	5.82	2.14
56	14	0	0.09	36.76	6.56	3.04
72	18	0.02	0.09	36.93	6.72	3.45
100	25	0.02	0.11	37.38	7.59	3.81

3.5 Summary

In this chapter, we propose to use consecutive drops for quick and accurate detection of inside packet drop attackers, design a new trust model that probabilistically penalizes attackers based on the consecutive drops, and build a hybrid trust model that adaptively uses the popular beta trust model and our new trust model. Extensive simulation results demonstrate that our hybrid trust model outperforms the beta trust model under various packet drop attacks in terms of detection speed, detection accuracy, routing reliability, and energy-efficiency.

There are several directions for future work. First, our results on network with lossy network show fairly large FAR. How to improve the accuracy of the proposed approach in such network is still a challenge. To address this problem, we study false alarm recovery problem in Chapter 4. Second, after the inside attackers become aware of our hybrid trust model, how they can respond to the challenge and launch more sophisticated attacks.

Chapter 4

FADER: False Alarm Detection and Recovery Mechanism

4.1 Overview

Trust-based routing approaches can gracefully mitigate insider packet drop attacks by building trusted paths to the destination. It is shown that trust-based routing improves the packet's successful delivery under insider packet drop attacks over routing algorithms that do not consider trust [12, 16, 22, 25, 71, 72, 73]. Clearly, the effectiveness of these trust-based routing protocols is based on their underlying trust models. A good trust model will help the routing algorithm to quickly and accurately identify inside packet drop attackers and find alternate routes to avoid them. However, as we have mentioned, it is challenging to distinguish the natural packet drops due to network problems from the intentional packet drops by the inside attackers. No trust model can detect all the inside attackers (100% detection rate) and not misclassify any of the good nodes (0% false alarm rate) [28]. While most of the existing approaches focus on how to improve the detection rate, we study the situation when the false alarm rate is high.

The rationale behind this study is as follows. First, natural packet drops do occur in WSN due to fading, interference, collision, noise, congestion, and others [4, 32]. In certain circumstances such as weather change or moving objects, there may be an abnormality in the WSN's operating environment which will cause non-negligible amount of packet drops and result in false alarms. Second, once a node is considered as untrustworthy or as an attacker, the existing trust-based routing mechanisms will not use this node any more. In the case of a false alarm, this will affect the routing performance.

Third, when a false alarm happens, a good node will be treated as an attacker or a node that goes down. This shortens the network lifetime as it is well-known that network's lifetime is closely correlated to the time when the first node in the network goes down [31, 39, 40].

Our main idea to treat false alarms in a trust-based routing is to give these nodes a second chance. They will still be involved in routing (but their participation will have little or no impact on the routing performance) so their neighbors can continue to monitor their behavior and evaluate their trustworthiness. If their neighbors are convinced that false alarm has occurred, these nodes will be re-considered as good nodes and play their normal role in routing. However, if these nodes are confirmed to be insider attackers, they will be eliminated from the routing table just like the current trust-based routing algorithms. We call this mechanism *False Alarm DEtection and Recovery*, or FADER in short.

The key contribution in FADER is that it allows a node to detect and correct (by recovery) its false alarm. Several questions need to be answered during the implementation of FADER: how to detect and recover all the false alarms quickly and cost effectively? how to ensure that an attacker will be detected as false alarm and recovered? how to satisfy the routing performance requirements such as packet delivery guarantees? how much can this help in routing performance and network lifetime? We will address these questions in the rest of this chapter.

In the rest of this chapter, we first discuss the false alarm problem in the current trust-based routing and outline our FADER approach in Section 4.2. In Section 4.3, we answer the above questions by elaborating the technical details of FADER. We report our

OPNET simulation setup and simulation results in Section 4.4. We conclude in Section 4.5.

Finally, we mention that the simulation validation of our approach is based on a simple but popular trust model, namely the beta trust model, and a greedy routing protocol, GPSR. However, the discussion of the design of FADER is generic for any trust model and any touring protocols. That is, the proposed FADER approach can be combined with a trust-based routing protocol based on any trust model to improve the routing performance and network lifetime.

4.2 False Alarm in Trust-based Routing

4.2.1 Recap: Trust-based Routing

We use the popular Greedy Perimeter Stateless Routing (GPSR) as an example to show how a trust mechanism can help to detect and avoid inside attackers.

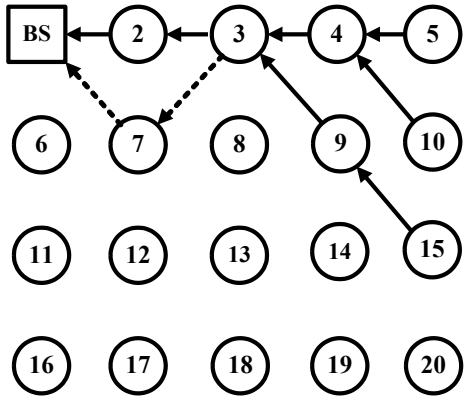


Figure 4.1: GPSR (Solid lines) and trust-based GPSR (Dotted lines). When node 2 drops many packets from node 3, a trust-based routing protocol will find a new routing path (Dotted lines) to avoid node 2.

Consider the WSN with 20 nodes shown in Figure 4.1. Node 3 relays the packets from nodes 4, 5, 9, 10, 15, and its own packets to node 2 who will then send them to the BS based on GPSR (depicted by solid lines with arrowhead). In a trust mechanism, node 3 will use its watchdog to monitor node 2.

When node 2 drops packets from node 3, node 3 will re-evaluate the trust value of node 2. If the trust value falls below the threshold value, node 3 will treat node 2 as an inside attacker. A trust-based routing protocol will then find a new routing path to avoid node 2. In this case, node 3 will forward the packets to node 7, hoping that node 7 will deliver the packets to the BS (the dotted lines with arrowheads in Figure 4.1).

4.2.2 Existence of False Alarms in Trust Models

As we have introduced in Section 2.5, the goal of a trust model is to provide a quantitative measurement of a node's trustworthiness. In existing trust models, a monitoring node A evaluates its neighbor node B based on how reliably B forwards the packets it receives from A. Because A cannot tell whether a given packet is dropped by B due to a network problem or maliciously, A may mistakenly claim B as untrustworthy when there are sufficient natural packet drops that occur in B. There are several reasons for this:

First, this may happen in a lossy network (or during the period when the network is lossy) when B drops many packets due to problems such as fading, noise, collision, congestion, or interference. These problems may be caused by network's uncertain environment and thus become unpredictable.

Second, the attackers inside the network can intentionally create packet drops at victim nodes by means such as jamming.

Third, with the desire of a high rate of detecting attackers, the trust threshold has to be set high. This comes at the cost of a high false alarm rate.

4.2.3 Impact of False Alarms on Trust-based Routing and Network

In current trust-based routing protocols, only trustworthy nodes are used for routing. Consider the same WSN in Figure 4.1. Node 3 considers node 2 untrustworthy and thus re-routes all its packets (as well as the packets from nodes 4, 5, 9, 10, and 15) to node 7. This will cause one or more of the following problems. When the network has a large number of false alarms, these problems can be critical to the network.

1) Degraded routing performance: No matter what the objective of the routing protocol is, because its original choice (the false alarm node) is replaced, the new routing path will not be as good as the original one. For example, the transmission distance and energy on path 3→ 7→ BS can be higher than those on path 3→2→BS.

2) Increased network traffic: Node 7 now has to relay the packets from additional 6 nodes (3, 4, 5, 9, 10, and 15) to the BS. This will increase the congestion, collision, and interference at node 7.

3) Reduced lifetime: with the increased workload at node 7, it will consume its resource (particularly energy) quicker than in the original routing solution. This shortens the lifetime of node 7. Furthermore, when node 7 goes down, other nodes (such as nodes

6, 8, and 12 in Figure 4.1) have to route packets from node 3. This will reduce their lifetime and quickly the lifetime of the entire network.

4.2.4 Overview of FADER

Clearly, the above problems are mainly caused by the false alarms and the action that the trust-based routing protocol takes on false alarms. Now since there is no trust mechanism that can completely eliminate false alarms, we need to change the way we treat the false alarmed nodes.

Recall that the existing trust mechanisms consider each node either trustworthy (**T**) of untrustworthy (**U**). A T-node B can become U-node if its trust value *T[B]* drops lower than the trust threshold *TH*. When this happens, the monitoring node A stops forwarding packets to B. Then, A stops monitoring it and updating its trust value. Therefore, a U-node cannot change to a T-node and be put back into a routing path. This can be summarized as a finite state machine (FSM) shown in Figure 4.2.

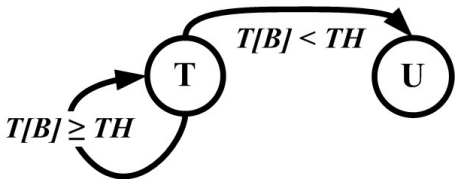


Figure 4.2: The 2-state finite state machine representation of current trust models. In this 2-state finite state machine, a U-node cannot change to a T-node.

From this FSM model, we see that if there is a false alarm U-node, we need an edge to transmit it back to a T-node. This is the basic concept behind our proposed False Alarm DEtection and Recovery (FADER) approach, where the key steps are outlined in

Figure 4.3. After node B's trust value *T[B]* is below the trust threshold *TH*, a new trustworthy node C is found to replace B. However, unlike the current approach, FADER keeps A monitoring B and updating B's trust value. This will provide the necessary data to determine whether B is an attacker or a false alarm. In the case of an attacker, A will permanently replace B by C in its routing table. In the case of a false alarm, A will use B again for routing and this is the way we make B a T-node which the current trust-based routing fails to do (see Figure 4.2). In the next section, we will elaborate in details of FADER, in particular with respect to the two shaded boxes in Figure 4.3: what data to collect and how to detect a false alarm.

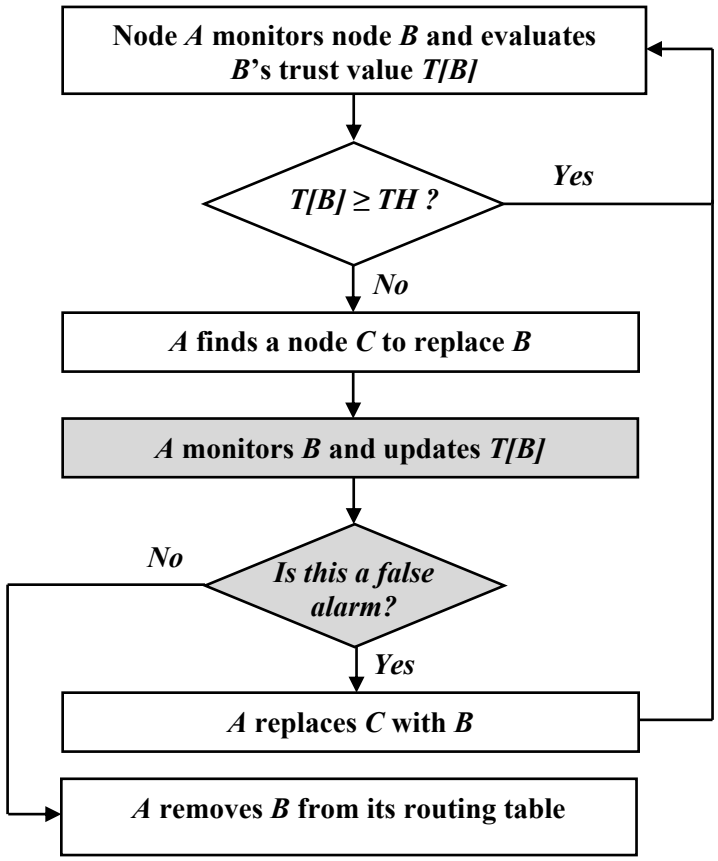


Figure 4.3: Outline of FADER: False Alarm DEtection and Recovery.

4.3 False Alarm Detection and Recovery Mechanism

In this section, we design a false alarm detection and recovery mechanism, which can be combined with existing trust models. We first design a 3-state trust mechanism with a recovery transition based on an FSM model, devise a trust re-evaluation method, and finally propose a multi-state trust mechanism with limited recovery chances that prevents malicious nodes from being recovered multiple times.

4.3.1 3-state Trust Mechanism with a Recovery Transition

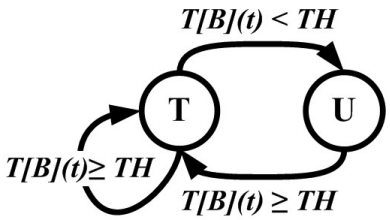


Figure 4.4: 2-state trust mechanism with a recovery transition. A U-node can change to a T-node unlimitedly when the recovery condition is satisfied.

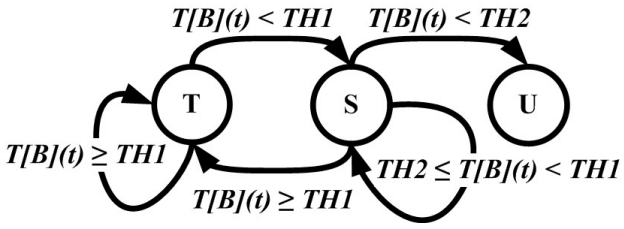


Figure 4.5: 3-state trust mechanism with a recovery transition. An S-node can change to a T-node, but a U-node cannot change to a T-node or a S-node.

To recover a falsely detected node in the state **U**, we need an edge from the state **U** to the state **S** so that the state transition **U** → **T** can be performed when a pre-defined

condition for recovery is satisfied. The existing 2-state trust mechanism may not be suitable as a recovery mechanism. As shown in Figure 4.4, we may simply add a recovery transition from the state **U** to the state **T**. However, in this model, it is not clear how to prevent extremely untrustworthy nodes (such as inside attackers) from being recovered. To address this, instead of 2-state trust mechanism, we use 3-state trust mechanism by adding an intermediate state **S** (Suspicious) between the state **T** and the state **U**, as shown in Figure 4.5. In this 3-state model, only the S-node can be recovered when it satisfies a pre-defined condition for its recovery. However, once an S-node becomes a U-node, it can never be recovered because there are no transitions such as **U** → **S** or **U** → **T**.

We explain our 3-state trust mechanism. First, we show how a monitoring node A classifies its neighbor node B into one of the three states (**T**, **S**, and **U**). Consider that A has been forwarding its data packets to B over the time *t*. A evaluates B's trust value $T[B](t)$ based on its direct observations on B's packet forwarding behavior over *t*. Note that $T[B]$ is measured by a trust model used in the network (e.g., by (2.5) when the beta trust model is used and by (2.7) when the entropy trust model is used). Given $T[B]$, two trust thresholds are necessary to classify B into one of the three states (**T**, **S**, and **U**). Let *TH1* be an upper trust threshold to separate two states **T** and **S**, and let *TH2* be a lower trust threshold to separate two states **S** and **U**. Then, given *T[B]*, *TH1*, and *TH2*, the state of node B at node A, $S_A[B]$, is defined as

$$S_A[B] = \begin{cases} T & for\ T[B] \geq TH1; \\ S & for\ TH2 \leq T[B] < TH1; \\ U & for\ T[B] < TH2. \end{cases} \tag{4.1}$$

Next, we explain how our 3-state trust mechanism works in each state transition between three states at the trust-based routing domain. As shown in Figure 4.5, we only use five state transitions as follows:

1) **T → T**: When a T-node B's trust value at A is equal to or higher than *TH1*, A continues to consider B as a trustworthy neighbor and forward its data packets to B.

2) **T → S**: When a T-node B's trust value goes below *TH1*, A considers B as an insider packet drop attacker, stops forwarding its data packets to B, and finds a trustworthy neighbor node C to forward its data packets to C. Meanwhile, based on our trust re-evaluation method that we will describe in the next section, A starts monitoring B to determine whether B is a falsely detected node (or a false alarm).

3) **S → U:** When an S-node B's trust value goes below *TH2*, A stops re-evaluating B, and discards B from its routing table. That is, A never forwards its data packets to B again.

4) **S → T:** When an S-node B's trust value goes above *TH1*, A considers that B was falsely detected and recovers B to a T-node. A replaces C with B as a current next hop (see Figure 4.3) and starts forwarding its data packets to B again. Meanwhile, A stops forwarding its packets to C. This is because B was A's original choice chosen by a base routing algorithm.

5) **S → S:** When an S-node B's trust value is between *TH1* and *TH2*, A continues to consider B suspicious and re-evaluates B to determine whether it is a false alarmed node or not.

Meanwhile, other unauthorized state transitions (such as **U → T** or **U → S**) must be prevented to avoid unauthorized recovery by malicious entities [55].

4.3.2 Trust Re-evaluation Method

Consider that node A wants to re-evaluate an S-node B to determine whether or not it is falsely detected. To do so, A needs some data that can be used for re-evaluating B's trust value. Obviously, the most reliable data for trust re-evaluation are A's direct observations on B's packet forwarding behavior to data packets that A sent to B. However, in existing trust-based routing protocols, it is impossible for A to get such direct observations on B, because once B becomes a U-node in A's trust mechanism, A never forwards its data packets to B. Instead of using direct observations on B, A may observe B's packet forwarding behaviors to other nodes and re-evaluate B's trust value based on the indirect observations. However, this approach is doubtable. Although B may forward another node C's packets successfully, there is no guarantee that B will also forward A's packets successfully. This is because the channel conditions between two different nodes are not identical.

On the other hand, our approach is that A *intermittently* sends its *duplicated* data packets to B for trust re-evaluation. There are three reasons why we use this approach. First, by this approach, A can directly observe B's packet forwarding behaviors to its own data packets and re-evaluate B's trust value reliably based on its own observations. Second, even if B drops a duplicated data packet that A sent to B for trust re-evaluation, the original packet will be delivered to the BS via a trustworthy neighbor C. Thus, this approach ensures that the overall packet delivery performance will not be degraded due to our trust re-evaluation. Last, A intermittently sends data packets to B for false alarm detection, because this approach will increase the possibility to recover a falsely detected

node under temporal packet transmission failures. In addition, this approach will have less communication overhead.

Based on this approach, we devise a simple trust re-evaluation method as follows. Whenever A starts re-evaluating B, A creates a random sequence $R(n)$ that indicates which of its data packets should be duplicated and sent to B for trust re-evaluation.

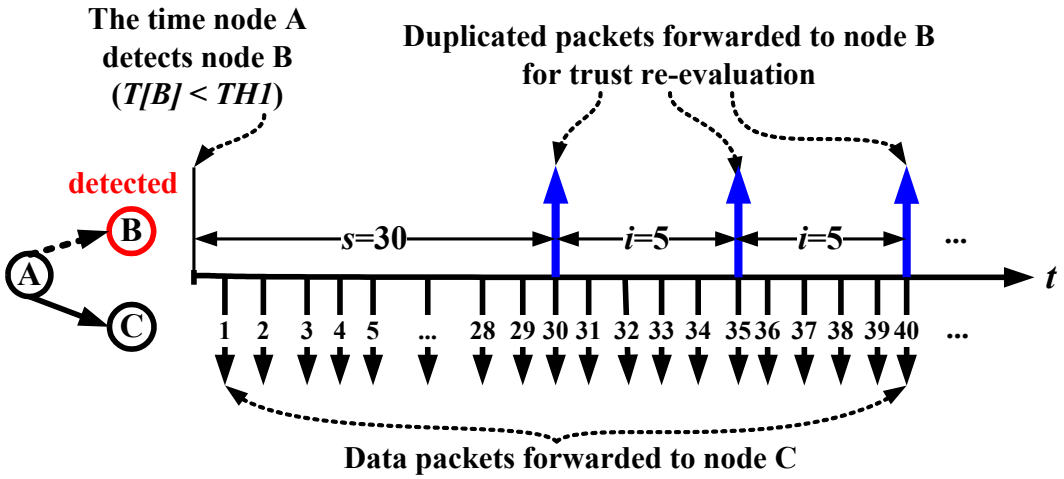


Figure 4.6: Our trust re-evaluation method based on a random sequence with s = 30 and i = 5. Node A will duplicate its 30th packet for the first time after node A detects node B as an untrustworthy node and send that packet to B for trust re-evaluation. After that, A will duplicate every fifth packet (35th, 40th, 45th, …) and send it to B until the trust re-evaluation is terminated.

As shown in (4.2) below, we design $R(n)$ by using two random functions $S(s)$ and $I(i)$ such that $S(s)$ determines the first value s of $R(n)$ and $I(i)$ determines the interval i between two consecutive values of $R(n)$. That is, i determines how frequently A will send a duplicated packet to B for trust re-evaluation; s and i are depicted in Figure 4.6.

$$R(n) = s + i \times (n-1) \quad for\ n = 1, 2, ..., \infty. \tag{4.2}$$

A simple way to design $S(s)$ and $I(i)$ is to use two uniform distributions. Before starting trust re-evaluation, A randomly chooses s in $[s_1, s_2]$ and i in $[i_1, i_2]$ where the parameters s_1, s_2, i_1 and i_2 are positive integers such that $s_1 < s_2$ and $i_1 < i_2$. For example, assuming that s and i are randomly chosen as $s = 30$ and $i = 5$, the generated random sequence $R(n) = \{5n + 25 | n = 1, 2, \ldots \infty\}$. Then, as shown in Figure 4.6, based on $R(n)$, A will duplicate its 30th packet for the first time after A detects B as an untrustworthy node and send that packet to B for trust re-evaluation. After that, A will duplicate every fifth packet (35th, 40th, 45th, …) and send it to B until the trust re-evaluation is terminated. In this method, the higher s and i are, the longer the trust evaluation period is.

4.3.3 Limiting Recovery Chances

In the 3-state trust mechanism, an S-node can be recovered to a T-node unlimitedly whenever its trust value goes above *TH1* (see Figure 4.5). As a result, this scheme can be vulnerable to a malicious node that changes its attack patterns like an on-off attacker. For this reason, we propose a multi-state trust mechanism that limits the number of recovery chances.

To limit the number of recovery chances, we further divide the state **S** into two sub-states: **SU** (Suspicious and Untrustworthy) and **ST** (Suspicious but Trustworthy). Figure 4.7 shows an FSM diagram of our multi-state trust mechanism with two recovery chances. In this mechanism, the state **S** is divided into two **SU** states (**SU1** and **SU2**) and two **ST** states (**ST1** and **ST2**). Like the 3-state trust mechanism, a node can be considered for recovery only when it is in state **SU** (that is, either in **SU1** or in **SU2**). However, unlike the 3-state trust mechanism, each node has only two chances to be recovered in

this mechanism. For example, consider that node A is re-evaluating node B that is in state **ST2** after spending two recovery chances. If B's trust value $T[B]$ goes below *TH1* again, B finally goes to the state **U** where B is completely discarded by A as shown in Figure 4.7.

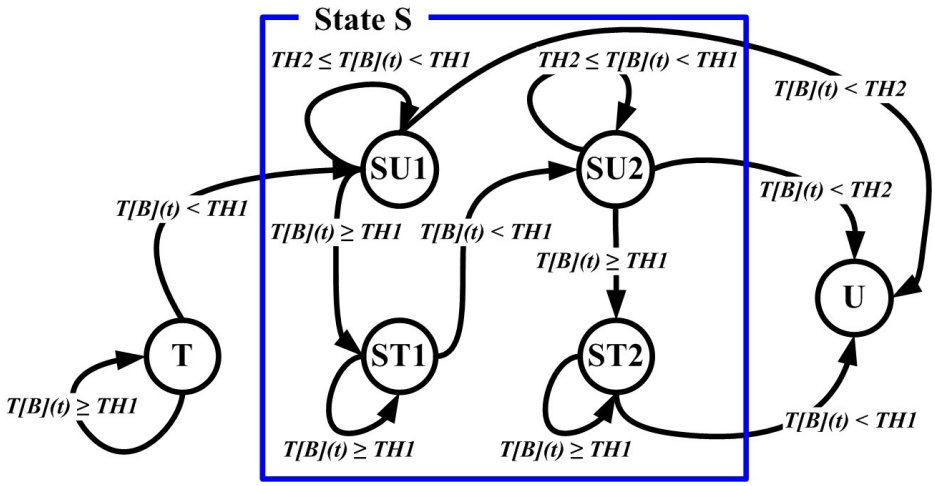


Figure 4.7: The multi-state trust mechanism with two recovery chances (recovery chance r=2).

For r recovery chances, we need r **ST** states and r **SU** states. The generalized restricted multi-state trust mechanism with i recovery chances is shown in Figure 4.8. Note that when the recovery chance $r = \infty$, such scheme will be the same with the 3-state trust mechanism without limiting recovery chances.

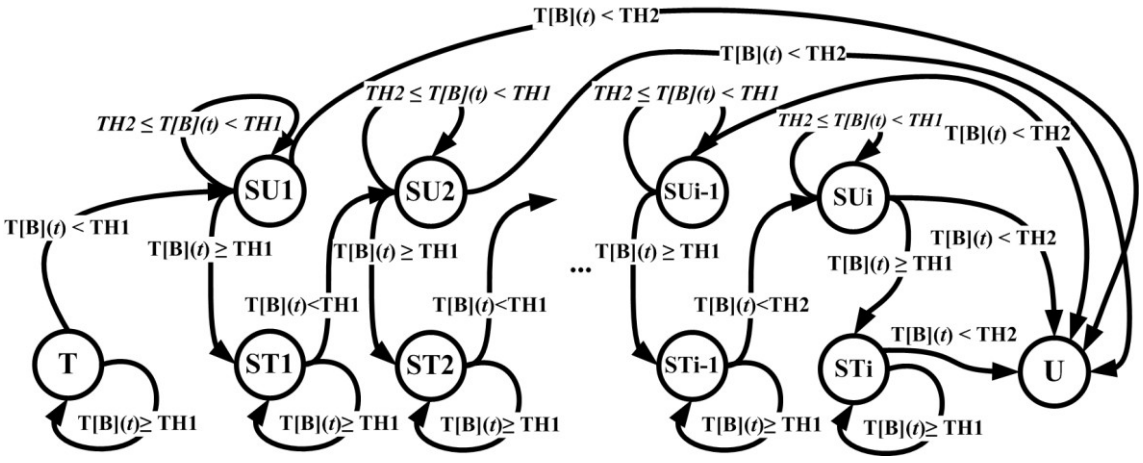


Figure 4.8: The multi-state trust mechanism with i recovery chances (recovery chance $r = i$).

4.4 Performance Evaluation

4.4.1 Simulation Goals, Setups, and Evaluation Metrics

The goal of our simulations is to evaluate the performance of the proposed FADER approach. Specifically, we consider a WSN with inside packet drop attackers and compare the performance of three routing protocols: geographic routing protocol (GRP), trust-based GRP based on Beta trust model (Beta GRP), and FADER GRP. We also study the accuracy and effectiveness of FADER in detecting false alarms.

We conduct simulations with the commercial network simulator OPNET Modeler under the following settings.

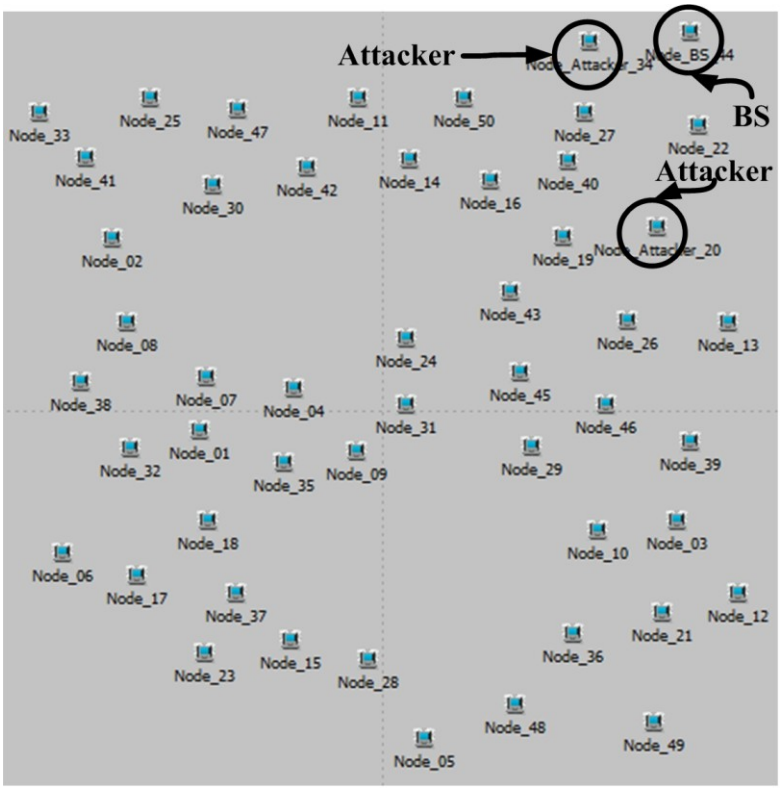


Figure 4.9: The ad hoc WSN with two inside packet drop attackers used in our simulations.

WSN setting: an ad hoc WSN with 50 sensor nodes is built randomly in a 1km×1km area as shown in Figure 4.9. We place the BS at one corner and every other sensor node will generate and send packets to BS. We use the default OPNET setting of data packet size (1KB) and GRP for routing. Each node generates a packet randomly within each 10-second period. We set the simulation time to be two hours (7,200 seconds). At the start of the simulation, each node has 30J of energy. To simulate the packet drops in a lossy network, we randomly choose *p*% of the nodes and let them drop packets during the period of [100s, 400s].

Attacker setting: we place two packet drop attackers near the BS. The attackers can launch any of the following attacks: blackhole attack (by dropping all the packets during the entire simulation period); grayhole attack (by randomly dropping each packet with a given probability); on-off attack (by repetitively dropping all packets for m seconds and then forwarding all packets for the next n seconds). In the last two cases, the drop rate is set to be higher than 1 – *TH1* (otherwise, the trust mechanism will fail to detect the attackers).

Trust model and FADER setting: The beta trust model as described in Section 2.5.3 is used. A trust-based GRP is implemented. Each sensor node's initial trust value is set to be 0.99. 0.8 is used as the trust threshold. The beta trust model may catch the two packet drop attackers as well as some of the *p*% of the nodes we selected to simulate lossy network. The latter ones will be false alarms. For FADER, 0.8 and 0.6 are used as *TH1* and *TH2*, and the number of recovery chances is set to be two. FADER starts the trust re-evaluation after 30 redundant packets are sent to a node whose trust value falls below *TH1*, and repeats the re-evaluation for every fifth packet.

We compare GRP, trust-based Beta GRP, and our proposed FADER in terms of the following performance metrics:

1) Network lifetime: we use two representative definitions of network lifetime: the simulation time when the packet delivery rate (PDR) drops dramatically (NL1), and the time when the first sensor node is drained of energy (NL2). A good routing protocol will give longer network lifetime.

2) Routing performance: we measure both the number of hops of each path and the average number of hops per path (AH). A good routing protocol will have fewer long paths and a smaller AH.

3) False alarm recovery rate: this is for FADER only. It measures FADER's ability to detect and recover false alarms from the beta trust model.

4.4.2 Simulation Results and Analysis

4.4.2.1 Improvement of Network Lifetime

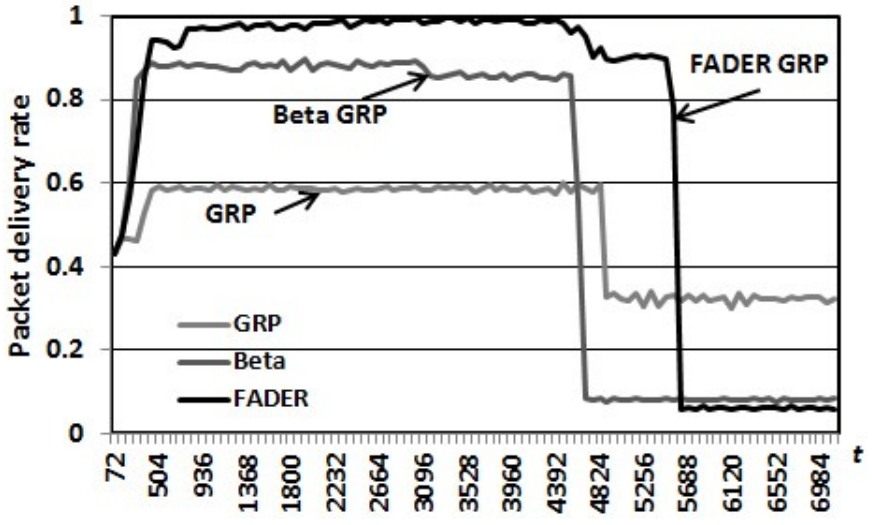


Figure 4.10: Packet delivery rate under two blackhole attackers when p = 8%. From bottom to top around time t = 1,800s: GRP, Beta GRP, and FADER.

Figure 4.10 reports the packet delivery rate (PDR) by three routing protocols for p=8%. In the case of GRP without any trust mechanism, all the packets that have an attacker on its routing path cannot reach the BS and thus it has a low PDR (slightly less than 0.6). At time around 4,824 seconds, sensor nodes start to run out of energy and the PDR drops dramatically from 0.6 to 0.3.

With the help of trust mechanism, we see that the PDR can be improved significantly to be around 0.9 in the case of trust-based Beta GRP and very close to 1.0 in the case of FADER. The better PDR performance of FADER is a direct result of false alarm recovery. We can also see that FADER has a longer network lifetime compared to Beta GRP.

The first node running out of energy happens (metric NL2) at time t = 3,162s for Beta GRP and t = 4,513s for FADER (see the last two columns of first row in Table 4.1). We can observe a very small drop in PDR because both routing protocols find alternate path to forward packets. However, as more and more nodes use their energy, routing paths become more and more difficult to find. Eventually, PDR drops dramatically when a critical node dies and no alternate path can be found (metric NL1: 4,536s for Beta GRP and 5,616s for FADER).

Table 4.1 reports more detailed data on network lifetime measured in terms of both NL1 and NL2. It includes two networks settings with p=8% and p=12%; and three different types of attackers. As expected, with a larger p value, trust mechanism will have more false alarms which cause much shorter network lifetime for Beta GRP. With the false alarm detection and recovery feature, FADER improves NL1 by 82.9% under

blackhole and on-off attacks. For grayhole attack, which is widely considered as the most challenging attack, the improvement is 54.3%.

Table 4.1: Network Lifetime (*NL1* and *NL2*): *NL1*: the simulation time when the packet delivery rate (*PDR*) drops dramatically; *NL2*: the simulation time when the first sensor node is drained of energy.

		NL1		NL2	
p(%)	Attacker type	Beta GRP	FADER	Beta GRP	FADER
8	Blackhole	4,536	5,616 (+23.8%)	3,162	4,513 (+42.7%)
	Grayhole(0.8)	4,608	5,688 (+23.4%)	3,177	4,595 (+44.6%)
	On-off(80, 20)	4,680	5,544 (+18.7%)	3,170	4,555 (+43.7%)
12	Blackhole	2,520	4,608 (+82.9%)	2,609	3,643 (+39.6%)
	Grayhole(0.8)	2,520	3,888 (+54.3%)	2,610	3,938 (+50.4%)
	On-off(80, 20)	2,520	4,608 (+82.9%)	2,620	3,653 (+39.4%)

FADER consistently extends network lifetime in terms of NL2 by about 43% over Beta GRP. Recall NL2 is the time when the first node uses up its energy and becomes inactive. In FADER, when we recover false alarms, which are the first and original choice of GRP, these recovered nodes normally give us a better routing path which takes less energy and smaller hops, and yield less congestion and collision. We will analyze this feature of routing performance in more details next.

4.4.2.2 Improvement of Routing Performance

We compare the routing solutions provided by the trust-based Beta GRP and FADER. First, each routing protocol will find a path for each node to reach the BS. Table

4.2 reports the number of the k-hop paths for each protocol under different values of *p* when the attacker drops 80% of packets randomly (Grayhole(0.8)).

Table 4.2: The number of k-hop paths in the routing solutions by Beta GRP and FADER when the attack type is Grayhole (0.8)

	p=0%	p=8%		p=12%	
k	Beta GRP	Beta GRP	FADER	Beta GRP	FADER
1	4	3	3	4	4
2	5	5	6	4	4
3	10	5	7	4	8
4	12	10	11	4	10
5	9	14	13	11	9
6	8	7	7	6	8
7	**1 (max)**	3	**1 (max)**	8	**6 (max)**
8	0	**1 (max)**	0	6	0
9	0	0	0	**2 (max)**	0
*∞	0	1	1	0	0
Total hops	192	209	194	253	211
AH	3.91	4.27	3.95	5.16	4.30

* $k = \infty$ means that the node cannot reach the BS.

At the second column in Table 4.2, the number of k-hop paths by Beta GRP is given when *p*=0%. When *p* grows to 8 or 12%, we can see that the Beta GRP has longer k-hop paths and longer average hops than the Beta GRP when p = 0% because of increased false alarms.

We see that FADER has the following advantages:

1) Fewer longer paths: when p=8%, Beta GRP has 4 paths of 7 hops or more, while FADER has only one 7-hop path. When p=12%, Beta GRP has 16 paths of 7 hops or more, while FADE has only six 7-hop paths.

2) Shorter longest path: the number of hops in the longest path is reduced by 1 and 2 for p=8% and p=12%, respectively.

3) Smaller average hop per path: as shown in the last row, AH is reduced by 7.5% and 16.7% for p=8% and p=12%, respectively.

In addition, these features imply that FADER provides a routing solution that (1) is more energy efficient (smaller average hops per path); (2) has less delay (shorter longest path); and (3) will cause less congestion (fewer long paths). Moreover, in the presence of false alarms, we can see that FADER works similarly with the Beta GRP when p = 0% by detecting and recovering false alarms.

4.4.2.3 False Alarm Recovery Performance

Finally, we discuss FADER's performance on detecting and recovering false alarms. First, we describe the metrics. Consider node 3 in Figure 4.1. It will receive packets from both nodes 4 and 9. Therefore, it will be monitored by both nodes 4 and 9. If node 3 is an inside attacker, a trust mechanism should enable both monitoring nodes to identify this.

For attacker $\boldsymbol{A_i}$, let $\boldsymbol{v_i}$ be the number of victim nodes that directly forward packets to $\boldsymbol{A_i}$; $\boldsymbol{s_i}$ be the number of nodes which successfully determine that $\boldsymbol{A_i}$ is an attacker; and $\boldsymbol{r_i}$ be the number of nodes that have identified the attacker but then mistakenly consider the attacker as a good node by FADER. Similarly, for each false alarm node $\boldsymbol{FA_i}$, we can

define fv_i and fs_i as well as fr_i which is the number of nodes that have generated a false alarm on node FA_i, but are able to detect the false alarm and recover by FADER.

Clearly, we want a trust mechanism to have s_i be the same as or as close as possible to v_i (that is, all victim nodes will identify the attacker) and fs_i be as small as possible (that is, fewer false alarms). When false alarms happen, we can measure FADER's false alarm recovery performance by comparing the values of fr_i and fs_i for each false alarm; and r_i and s_i for each attacker. We define FADER's recovery rate $RR = \sum fr_i / \sum fs_i$ where the sum is taken over all the false alarms.

Table 4.3: False detection recovery performance: The results show that FADER is able to recover 60−70% of the false alarms without recovering any of the attackers in the presence of multiple attackers in lossy WSNs.

	Attack	Beta GRP		FADER		
		$\sum v_i = \sum s_i$	$\sum fs_i$	$\sum r_i$	$\sum fr_i$	RR
p=8%	Blackhole	5	10	0	7	70.0%
	Grayhole(0.8)	5	10	0	7	70.0%
	On-off(80, 20)	5	10	0	7	70.0%
p=12%	Blackhole	5	14	0	10	71.4%
	Grayhole(0.8)	5	15	0	10	66.7%
	On-off(80, 20)	5	15	0	9	60.0%

Table 4.3 reports the recovery performance of FADER. First we notice that the beta trust model is able to let all victims nodes identify the attack ($\sum v_i = \sum s_i$), however, it does create a considerable amount of false alarms ($\sum fs_i$), which is about 2-3 times of $\sum v_i = \sum s_i$. This is the direct consequence of seeking for a high attacker identification rate.

With FADER, we see that it does not recover any attacker ($\sum r_i$=0 for all cases), while it successfully detects and recovers most of the false alarms ($\sum fr_i$) with at least 60% recovery rate in ***RR***.

Finally, we mention that FADER's high false alarm recovery rate helps us to obtain the improvement of network lifetime and routing performance as analyzed earlier.

4.5 Summary

Trust-based routing is critical for WSNs when there may exist inside packet drop attackers. To ensure high PDR, it is desirable for a trust mechanism to catch all the inside attackers. However, high false alarms may cause significant degradation of network lifetime and routing performance. To our knowledge, FADER is the first attempt to address the false alarm problem in trust-based routing. We propose a false alarm detection and recovery (FADER) approach which enables us to identify and reuse these false alarmed nodes. Extensive OPNET simulations confirm that the FADER approach can dramatically improve network lifetime, package delivery rate, and the performance of the routing.

There are several directions for future work. First and perhaps the most challenging question is how the inside attackers will respond to FADER. Because inside attackers are legal members of the network, they will know the details of the FADER and launch sophisticated attacks. Second, as we see in the results, FADER can recover about 70% of the false alarms, and there is still room to improve the accuracy of FADER. Finally, it will be interesting to test FADER on more networks with more realistic noise model and other trust models.

Chapter 5

Detection and Prevention of Selective Forwarding-based Denial-of-Service Attacks

5.1 Overview

Selective forwarding attack, where the attacker drops only some packets and at some arbitrary time, is the most difficult insider packet drop attack to defend against [7]. Normally such an attacker seeks to achieve one of the following two goals. First, degrade the performance of the network in terms of packet loss rate. Second, prevent data collected by certain sensor nodes from reaching the BS. In the second case, the victim node will not be able to talk to the BS. We name this attack *selective forwarding-based denial-of-service (DoS) attack.* As we discussed in Section 2.3, most reported detection approaches on selective forwarding attacks focus on the detection of the attacker with the first goal, and thus they are not effective against selective forwarding-based DoS attacks [4, 9, 17, 18, 47, 50, 60]. Existing avoidance approaches based on multipath routing [7] or multiple topologies [5] introduce high communication overhead, and they are vulnerable to multiple attackers [4].

As a motivation for the importance of studying selective forwarding-based DoS attacks, we consider a WSN deployed in a territory for intruder detection. With the help of insiders that perform the selective forwarding-based DoS attack, an intruder will be able to enter the territory from the area monitored by victim nodes (to the selective forwarding-based DoS attacks) without being noticed by the BS. When the intruder can

communicate with the inside attackers, they can launch the *synchronized insider-outsider colluding DoS attack* so the insider attackers can target different victims at different times and the intruder can explore the territory covered by the victim nodes only.

In this chapter, we study the selective forwarding-based DoS attacks and propose effective detection and avoidance mechanisms as well as a prevention routing algorithm to defend against such attacks. Specifically,

- we first describe a simple selective forwarding-based DoS attack and show that the popular trust-based approaches (such as beta and entropy trust mechanisms) for inside attacker detection fail to detect such attack. We also analyze the potential damage this attack can cause to the network.
- we then propose a *source-level trust evaluation scheme* to enhance the beta and entropy trust mechanisms for effective detection of the selective forwarding-based DoS attackers. Once the attacker is identified, we propose two *avoidance strategies* to re-route the victim's packets so they can reach the BS.
- we validate our claims and evaluate the performance of our detection and avoidance mechanisms with extensive OPNET simulations.
- as a complementary defensive mechanism to our detection and avoidance methods, we also introduce a prevention routing algorithm to proactively prevent the selective forwarding-based DoS attacks and show our preliminary results to evaluate its performance.

For simplicity, during the discussion of the threats and detection of insider packet drop attacks, we do not consider natural packet drops caused by network problems.

However, our simulation settings include lossy networks and the natural packet drops due to network problems will be reported.

The rest of this chapter is organized as follows. We first describe a selective forwarding-based DoS attack that none of the current defending approaches can detect to motivate our work in Section 5.2. We propose our detection and avoidance approaches in Section 5.3 and evaluate their performance in the packet routing domain in Section 5.4. In Section 5.5, as a complementary defensive mechanism to our detection and avoidance methods, we introduce a prevention routing algorithm where an attacker has to choose between "not attacking" and "attacking and being caught". We summarize in Section 5.6.

5.2 A Selective Forwarding-based DoS Attack and Its Analysis

1) Motivation: the current trust mechanisms and trust-based routing cannot detect all known insider packet drop attacks. For instance, an intelligent attacker who can keep its trust value above the threshold value $\boldsymbol{\Theta_T}$ will not be detected. More weakness can be found in literature such as [35]. Our proposed selective forwarding-based DoS attack comes from the following simple observation:

To attack victims and avoid being identified, the attacker node ***A*** will have to disguise itself by forwarding packets for some nodes. When a node ***M*** sends only its own packets to the attacker ***A*** and uses its watchdog to monitor ***A,*** apparently ***A*** cannot drop all the packets without being detected. However, if ***M*** also forwards packets from other nodes to ***A***, then ***A*** maybe able to drop all the packets from one or multiple victim nodes.

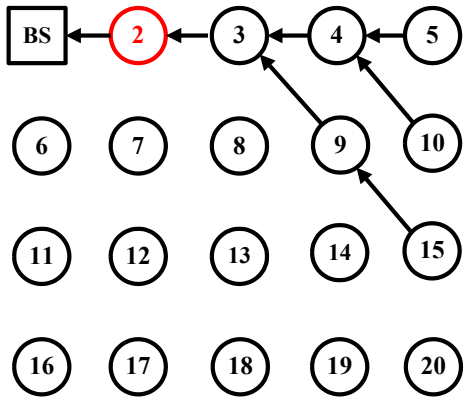


Figure 5.1: Trust-based GPSR in the presence of a selective forwarding-based DoS attacker (node 2).

For example, in the WSN shown in Figure 5.1 where all the nodes generate packets with the same frequency and send them to the BS, node 2 can pick node 10 as its victim and drops all the packets from node 10. Therefore the BS can never hear messages from node 10 and hence comes the name "denial of service" for this attack. However, if node 2 forwards all the packets from nodes 3, 4, 5, 9, and 15 to the BS, when Beta trust model is used with $\boldsymbol{\Theta_T}$ =0.70, the monitoring node 3 will hear node 2 forwarding 5/6 ≈ 83% of the packets and fails to identify node 2 as an attacker because node 2's trust value will be approximately 0.83, higher than the threshold $\boldsymbol{\Theta_T}$ =0.70.

It is not hard to see that once an attacker position itself on the routing path of many nodes, it can select multiple victim nodes and launch the denial of service attack without being noticed. This can easily cause a lot of damage to the network and so we need to find countermeasures to defeat such attack.

2) Protocol of the attack: the following steps define the protocol for an inside attacker $\boldsymbol{A}$ to launch the selective forwarding-based DoS attacks against multiple victim nodes:

1. for each node ***M*** that forwards packets directly to ***A*** and monitors ***A*** with its watchdog, $k_M = 0;\ n_M = 0;$

2. while (both the network and node ***A*** are on) {

3. on the reception of a packet {

4. identify the node ***M*** that forwards the message;

5. identify the source node S that generates the packet;

6. if S is a victim node, drop the packet;

7. if S is a non-victim source node, forward the packet;

8. if S is a new source node {

9. n_M++;

10. if $n_M = V[k_M]$ {

11. pick a new victim source node;

12. k_M ++;

13. if S is the new victim node, drop the packet;

14. else forward the packet;

}

}

}

}

On each received packet (step 3), the attacker ***A*** first determines the direct sender of the message (node M) and original source node S that generates the packet (steps 4 and 5). If ***A*** has received packets from S before (that is, S is not a new source node), ***A*** will either drop or forward the packet based on whether S is a victim or not (steps 6 and 7). If

S is a new source node, $\boldsymbol{A}$ will update the number of nodes whose packets are routed to $\boldsymbol{A}$ through $\boldsymbol{M}$ by $\boldsymbol{n_M}$++ (step 9). When $\boldsymbol{n_M}$ reaches a pre-determined value, $\boldsymbol{A}$ will be able to select a new victim to launch the DoS attack (steps 10-14). We called this attack *selective forwarding-based DoS* because the attacker can selectively choose the victims and drop all the packets from the victims to mislead the BS to consider the victim nodes are either out of service or disconnected.

3) Analysis of the attack: for an inside attacker to launch the selective forwarding-based DoS attack against the victim nodes, the attacker needs to (i) be able to tell whether a received packet is from the victim nodes, and (ii) ensure that after dropping all the packets from the victim nodes, the attacker will not be detected by the monitoring nodes.

We first show that assumption (i) is valid. In a geographic routing employed WSN, the receiver of a packet can obtain the source node (the node that creates the packet) information from the packet, because the receiver is a legitimate relay node that can access the packet's header where the source identification is stored [8, 69]. Even when the source node is protected by methods such as authorization, it is still possible for a malicious receiver to figure out the source node information by breaking the authorization mechanism or analyzing network traffics [36, 37].

Second, we will show requirement (ii) can be satisfied. Because the inside attacker is a legitimate member of the WSN, it knows the trust model and the threshold value $\boldsymbol{\Theta_T}$ used in the network. In a well-defined trust mechanism, when a node's packet drop rate increases, its trust value should not increase. A node will be considered as trustworthy if its trust value is above the threshold. Therefore, an attacker can evaluate its own trust value and drop a packet only when a drop will not bring its trust value below

the threshold Θ_T. In our proposed protocol, the attacker selects a victim only when there are enough non-victim source nodes to keep the attacker's trust value above Θ_T. This is guaranteed by the carefully determined array ***V[.]*** used in step 10 as we will explain next.

We define ***V[j]*** as the minimum number of source nodes whose packets are routed to the attacker (***A***) through the same monitoring node (***M***) such that the attacker can drop packets from ***j+1*** of these nodes without being detected by ***M***. That is, ***V[0]*** is the minimum number of nodes for attacker ***A*** to cover/disguise the first victim; ***V[1]*** is the minimum number of nodes for ***A*** to attack two victims.

In the beta trust model, if attacker ***A*** attacks (j+1) victims among ***V[j]*** nodes and forward the packets for the other (***V[j]***-(j+1)) nodes, its trust value will be

$$\frac{V[j]-(j+1)}{V[j]} = 1 - \frac{j+1}{V[j]}$$

To keep this trust value higher or equal to the trust threshold Θ_T, we can easily obtain the following

$$V[j] = \left\lceil \frac{j+1}{1-\theta_T} \right\rceil$$

For the entropy trust models, there is no closed formula for ***V[j]***. However, we can compute ***V[j]*** numerically for any given Θ_T. Table 5.1 lists the values of ***V[j]*** for the three different trust models where 0.70 is used as the trust threshold Θ_T.

Table 5.1: Values of *V[j]* for three trust models when $\Theta_T = 0.70$

V[j]	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
T_{Beta}	4	7	10	14	17	20
T_{Entr}	19	38	57	76	94	113
T^*_{Entr}	7	14	21	28	35	42

The small values of ***V[0]*** indicate that the proposed selective forwarding-based DoS attack is a very serious threat. For the attacker (***A***) to launch the attack against a specific victim (***V***), it only requires the node (***M***) that forwards ***V***'s packets to ***A*** also forwards packets from 2 other nodes to ***A*** in the beta trust model (***V[0] = 4: M, V,*** and 2 other nodes).

One can also see that ***V[j]*** has a much larger value for the entropy trust models than the beta trust model. This is because earning a high trust value in entropy trust models (2.7) and (2.8) is much harder (that is, a node must have very few packet drops) than earning a high trust value in the beta trust model (2.5).

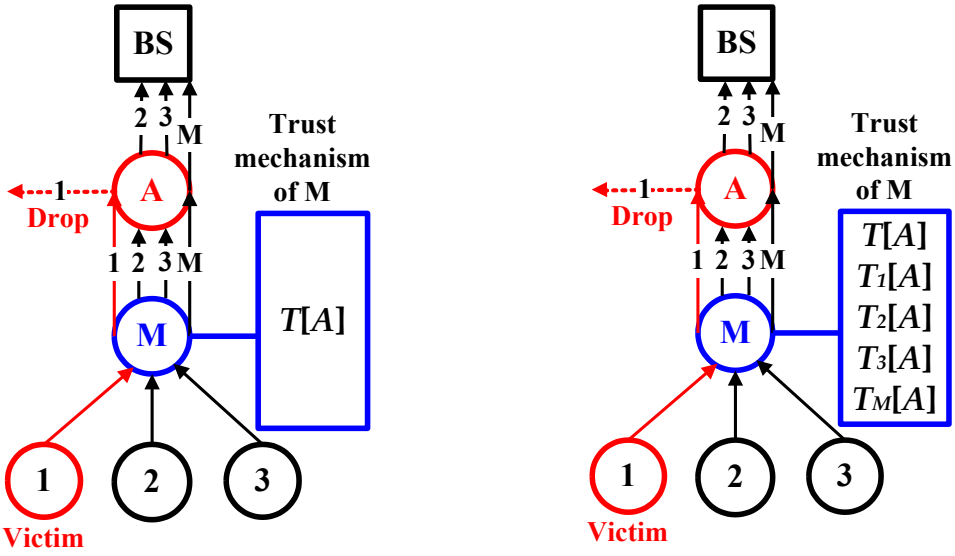
5.3 The Proposed Defensive Mechanism

In this section, we propose and analyze our defensive mechanism, which is an enhancement of the beta and entropy trust mechanisms, against the above selective forwarding-based DoS attack. This defensive mechanism consists of two phrases: attacker detection and attacker-aware re-routing, which will be elaborated in subsections 5.3.1 and 5.3.2 of this section, respectively. We analyze our approach and compare with existing methods in subsection 5.3.3.

5.3.1 Source-level Trust Evaluation and Attacker Detection

As depicted in Figure 5.2(a), in the existing trust mechanism [6, 10, 12, 22, 25, 71, 72, 74], a monitoring node M counts the number of successes s and failures f that the next node A forwards packets from M. It then evaluates the trust value $T[A]$ of A based on s and f using the trust model adopted by the network. If $T[A] < \boldsymbol{\Theta_T}$, M will consider A as

an inside attacker. However, we have seen that this mechanism fails to detect intelligent attackers such as those launching the selective forwarding-based DoS attacks. For example, attacker A can drop all packets from node 1, but forwards packets from all other non-victim nodes (in this case, nodes 2, 3, and M) to keep its trust value $T[A]$ high. When $T[A] \geq \boldsymbol{\Theta}_T$, A's malicious attacking behavior will not be detected by M.



(a) Existing approach

(b) Source-level trust evaluation

Figure 5.2: Existing trust evaluation approach and proposed approach. In our approach (source-level trust evaluation), M evaluates not only A's overall trust value T[A] but also A's source-level trust values T_i[A] to see how much M can trust A in forwarding packets from node i.

We can see that the current trust mechanism fails because the attacker can hide its malicious behavior behind its good behaviors. As an attacker can identify the source node of a packet to launch the selective forwarding-based DoS attack, a monitoring node can also utilize the source node information to defend against such attack. This leads us to the following idea. If M uses separate counters to track not only A's overall packet forwarding behavior, but also how it delivers packets from each individual source node,

then M will be able to tell whether A has launched the DoS attack against any node. This is shown in Figure 5.2(b) where M also evaluates A's trust value $T_i[A]$ for each source node i. We refer to this approach as s*ource-level trust evaluation* and it can be easily integrated into the current 3-stage trust mechanism to improve its effectiveness of detecting inside attackers as follows:

1) Neighbor behavior monitoring: In addition to recording A's overall behavior s and f, for each packet that M overhears A is forwarding, M checks the source node information and updates a pair of separate counters, s_i and f_i, where i is the source node of the packet, to keep tracking the number of successes and failures for packets that A forwards from source node i, according to A's packet forwarding behaviors to node i.

2) Trust measurement: Based on the data collected in the first stage, M evaluates not only A's overall trust value $T[A]$ based on s and f, but also its source-level trust values $T_i[A]$ based on (s_i, f_i) to see how much M can trust A in forwarding packets from node i. When the beta trust model is used, A's source-level trust value for source node i, $T_{Beta,i}[A]$, can be calculated by using (2.5) as

$$T_{Beta,i}[A] = \frac{s_i + 1}{s_i + f_i + 2} \tag{5.1}$$

When the entropy trust model is used, A's source-level trust value for source node i, $T_{Entr,i}[A]$, can be calculated by using (2.7) as

$$T_{Entr,i}[A] = \begin{cases} 1 - H(p_i) & for\ 0.5 \le p_i \le 1; \\ H(p_i) - 1 & for\ 0 \le p_i < 0.5 \end{cases} \tag{5.2}$$

where $H(p_i) = -p_i \log_2 p_i - (1-p_i)\log_2(1-p_i)$ and $p_i = (s_i+1)/(s_i+f_i+2)$. To have a non-negative trust value between 0 and 1, we define

$$T^{*}_{Entr,i}[A] = \frac{1 + T_{Entr,i}[A]}{2} \tag{5.3}$$

3) Attack Detection: If any trust value $T_i[A]$ goes below the pre-determined trust threshold, M detects that A is a selective forwarding attacker against node *i*, the victim of such attack. When the overall trust value $T[A]$ of node A goes below the trust threshold $\boldsymbol{\Theta_T}$, A will be considered as an inside packet drop attacker just like the current trust mechanism will do.

Theorem 5.1. The proposed source-level trust evaluation approach can successfully detect selective forwarding-based DoS attacks against any source node.

Proof. By the definition of the selective forwarding-based DoS attack, if A launches attack against node *i*, it will behave like a blackhole attacker and drop all packets originated from node *i*. Hence, after the attack is launched, s_i will remain unchanged and f_i will increase by one whenever a packet from node *i* is dropped by attacker A. When node *i* generates sufficient number of packets, the packet drop rate $\alpha = f_i/(s_i+f_i)$ will increase and can be arbitrarily close to 1. This means that A's trust value with respect to node *i*, $T_i[A]$, will approach to the minimum trust value, which will be way below the trust threshold $\boldsymbol{\Theta_T}$. So the monitoring node M will be able to identify this DoS attack and its victim.

Formally, let $n_i = s_i + f_i$ be the total packets generated by a victim node *i*, this theorem is based on the following fact.

$$\lim_{n_i \to \infty} T_i[A](\alpha) = \lim_{n_i \to \infty} T_i[A](f_i/(s_i + f_i)) = T_i[A](1) << \theta_T .$$

Because all the (s_i, f_i) pairs are kept independently, the selective forwarding-based DoS attack against any other source nodes can also be detected, depending on how fast the victim nodes generate packets. ■

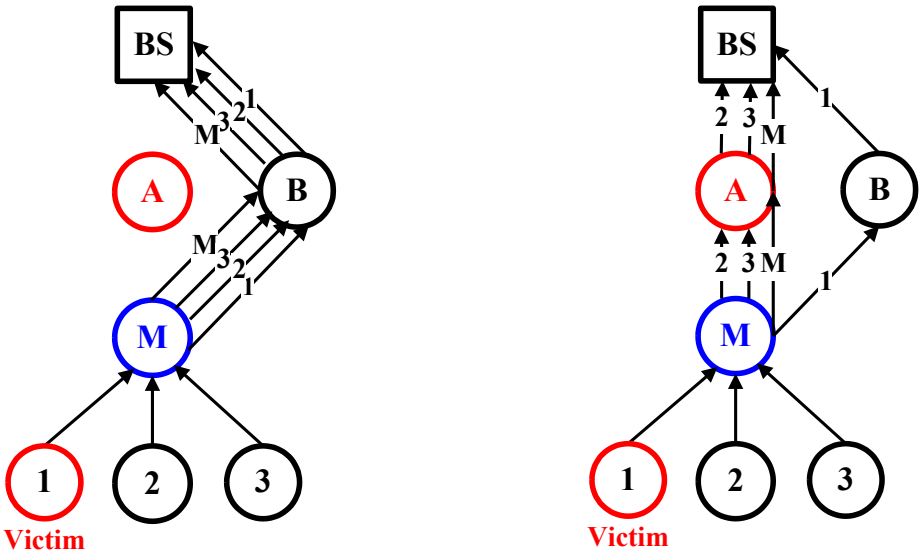
Our approach requires the number of delivery successes and failures for packets from each source node. This will introduce storage overhead. Fortunately, such overhead is negligible. Even in the case when a node is receiving packets from 100 different source nodes and wants to track the status of the last 1 million packets from each node, the memory requirements will only be 0.25KB ($=100\times\log_2 2^{20}$ bits/8). This overhead is low for current sensors such as TelosB (10KB RAM, 48KB Flash, and 1MB EEPROM) and Mica2/MicaZ (4KB RAM, 128KB Flash, 512KB EEPROM) [38].

5.3.2 Attacker-aware Avoidance Routing Strategies

Once the attacker and a victim of the selective forwarding-based DoS attack are detected, approaches to re-route the victim's packet to the BS should be developed. In this section, we propose two attacker-aware re-routing algorithms, which we refer to as avoidance strategies.

When the value of a $T_i[A]$ becomes less than the network's trust threshold $\boldsymbol{\Theta_T}$, the monitoring node M will conclude that A is an inside attacker attacking node *i*. To avoid further damage that A may make to the network, M can use a *complete avoidance* (CA) strategy to re-route all the packets to another trustworthy neighbor node (such as B shown in Figure 5.3(a)). This ensures that all the packets received by M, not only those from node *i*, will avoid the attacker A. However, this strategy will increase the traffic on node

B and may also introduce other routing overhead. For example, if node A was the best choice in an energy efficient routing algorithm, re-routing all the packets to node B instead of A will cause increase in energy consumption. Furthermore, if A targets multiple victims, this strategy will help all of the victims to avoid the attacker A, but it can only identify the first victim. Finally, assuming that node A is a good node, if M mistakenly claims the first victim, A will be treated as an attacker. This will increase the false alarm rate in finding inside packet drop attackers.



(a) Complete avoidance (CA)

(b) Selective avoidance (SA)

Figure 5.3: Two avoidance strategies to re-route the victim's packets to BS: CA: Complete Avoidance; SA: Selective Avoidance.

In light of the fact that a selective forwarding-based DoS attacker (node A in this case) has targeted victims, the *selective avoidance* (SA) strategy will only re-route the discovered victim's packets to avoid the attacker A and keep the other packets running through node A (see Figure 5.3(b)). The monitoring node M will continue updating the

trust values ($T_i[A]$) for all nodes except those discovered victims. So even when the attacker targets multiple victims, the SA strategy can discover all of them and help them to avoid the attacker. This strategy will effectively solve CA's resource overhead problem. Its drawback is that it will take time for each of the victims to be identified and the attacker can still drop packets from the victims and do damage to the network until all the victims are discovered.

We summarize the features of the two proposed attacker-aware re-routing algorithms in the following table:

Table 5.2: Comparison of the two proposed avoidance strategies (CA: Complete Avoidance; SA: Selective Avoidance)

	CA	SA
re-route victim's packets	Yes	Yes
re-route non-victim's packets	Yes	No
time to re-route all victims' packets	Short	Long
discover multiple victims	No	Yes
probability of false alarm on attacker	Large	Small
impact on the original routing solution	Large	Small

5.3.3 Analysis of the Proposed Defensive Mechanism

The proposed defensive mechanism follows the 2-phase detection-avoidance framework. In the first phase, the source-level trust evaluation approach will detect victims of the selective forwarding-based DoS attack. In the second phase, the attacker-aware re-routing strategy will find a different path to deliver victim's packets to the BS.

5.3.3.1 Comparison with the Existing Trust Mechanisms

Our source-level trust evaluation is an enhancement of the existing trust-based mechanisms for inside attacker detection [6, 12]. The difference is that existing approaches do not consider the packet forwarding behavior of the receiving node (the node being monitored) for each individual source node. Therefore, it can detect whether the node is an inside packet drop attacker, but it will fail to detect the proposed selective forwarding-based DoS attack. In our proposed method, the monitoring node will evaluate the trust value with respect to each source node. As stated in Theorem 5.1, this enhancement enables us to identify not only the attacker, but also all the victims. The cost of our approach, compared with existing mechanisms, is the storage requirement to keep the delivery information for each source node, which we have analyzed after the proof of Theorem 5.1.

Now we compare the false alarm rate (FAR) of our approach with existing mechanisms. FAR measures how likely a good node will be tagged as an inside attacker. Let $\boldsymbol{FAR}$, $\boldsymbol{FAR_{CA}}$, and $\boldsymbol{FAR_{SA}}$ be the FAR of the existing detection approach, our approach with CA, and our approach with SA, respectively. We have

Theorem 5.2. $\boldsymbol{FAR_{CA} \geq FAR \geq FAR_{SA}}$.

Proof. Recall that the trust value $T[A]$ in the existing trust mechanism is defined based on the packet drop rate, which is the ratio of the total failures (f) over the total number of packets ($s+f$). A false alarm occurs when a good node's trust value $T[A]$ becomes smaller than the trust threshold Θ_T. In our approach, the monitoring node M also

updates $T_i[A]$, the trust value with respect to source node i, which is determined by the drop rate of packets from node i or the pair of (s_i, f_i).

When we use CA strategy in the second phase, the first detected victim node j is the one that has the largest packet drop rate that results in the smallest $T_i\ [A]$ among all the node i's that send their packets to A through the same monitoring node M. That is, $T_j[A] = min\ \{T_i[A]\}$. For the same set of node i's, we have $s=\sum s_i$ and $f=\sum f_i$. Clearly, $T[A] \geq T_j[A]$. Therefore, when the existing detection mechanism claims (regardless the correctness of the claim) node A as an attacker, (that is, $T[A]<\Theta_T$), our approach should have already identified the first victim j of A's DoS attack because

$$T_j[A] \leq T[A] < \Theta_T.$$

However, when our approach claims an attacker, $T_j[A] < \Theta_T$, it is not necessarily true that $T[A] < \Theta_T$. A false alarm is an incorrect claim. So $\boldsymbol{FAR_{CA} \geq FAR}$.

On the other hand, when SA strategy is applied, our approach will identify the DoS victims one by one and re-route the packets from these victims to nodes other than the attacker A. Note that victim nodes always have large packet drop rate, when their packets are re-routed, the trust value of A evaluated by our approach will be higher than that in the existing approach. This is because the existing approach will count the (s_i, f_i) pairs from these victims in $s=\sum s_i, f=\sum f_i$ in the calculation of $T[A]$. So when the same Θ_T is used, $T[A] < \Theta_T$ will always first happen in the existing approach before it happens in our approach with SA strategy, that is, $\boldsymbol{FAR \geq FAR_{SA}}$.

5.3.3.2 Comparison with the Avoidance Approaches

As we have mentioned in the introduction, the idea behind current avoidance approaches is to send packets from multiple disjoint paths in order to avoid inside packet drop attackers [5, 7, 58, 59]. These approaches cannot and are not intended to detect the attackers. We have also discussed in Chapter 2 that the overhead of such avoidance approaches can be prohibitively high. For example, when each packet is sent through multiple different paths, the transmission energy, the network traffic, and collision will all increase dramatically.

Despite the same name, the avoidance strategy in the second phase of our defensive mechanism is conceptually different from the above avoidance approaches. In our approach, the avoidance strategy is applied after both the victims and the attacker in the insider packet drop attack have been identified. Therefore we can efficiently find a path that does not involve the attacker to deliver victim's packets to the BS. Although the new path may not be as good as the initial path (where the attacker sits on) in terms of energy, delay, or channel quality, neither CA nor SA uses multiple paths. Hence, the large overhead problem in the conventional avoidance approaches does not exist in our defensive mechanism.

5.4 Simulation and Results Analysis

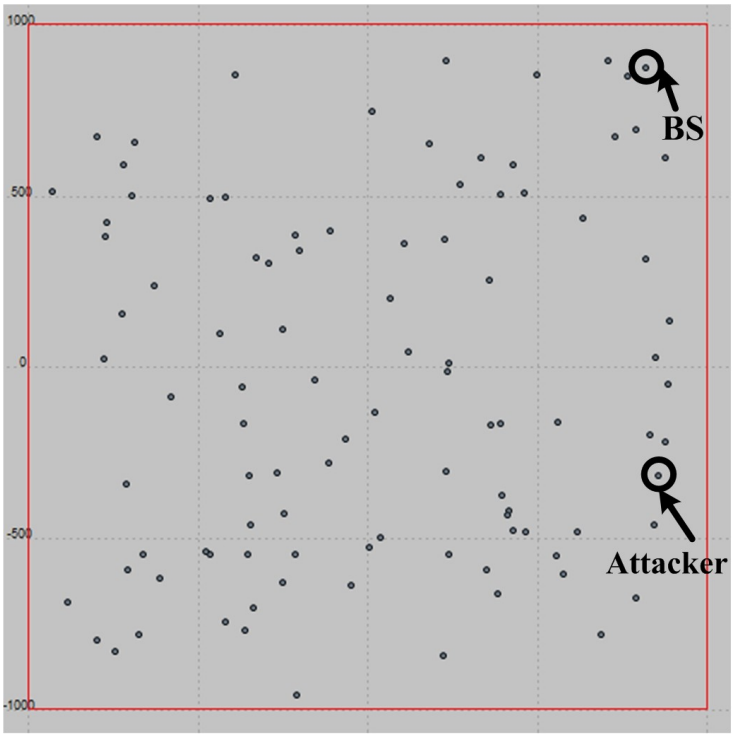
5.4.1 Simulation Goals, Setups, and Evaluation Metrics

There are two main goals of the simulation: validating that the current trust mechanisms fail to detect the proposed DoS attack; evaluating the performance of our defensive approach.

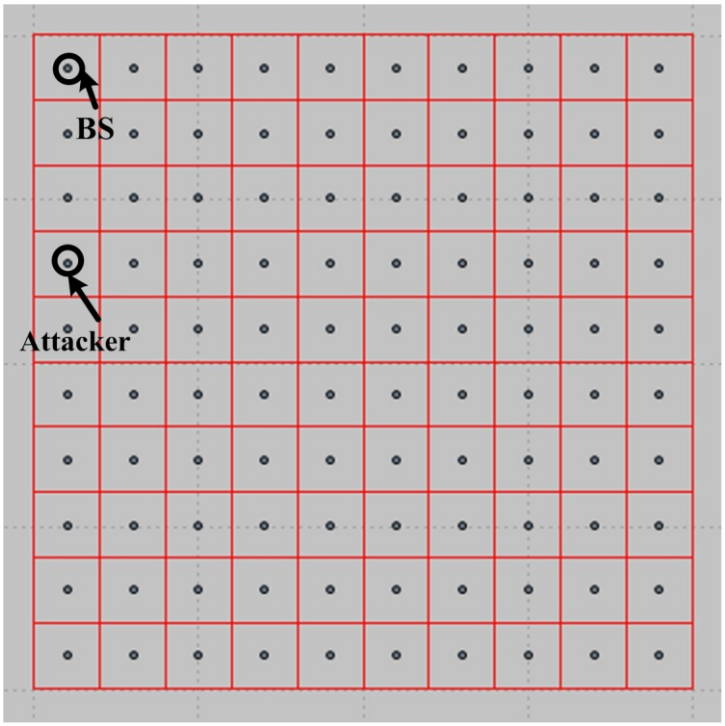
The parameters in Table 5.3 are used in our simulations. We conduct simulations with the commercial network simulator OPNET Wireless Modeler v.17.1. 100 sensors are deployed in a 2km×2km area randomly in one setting (Figure 5.4(a)) and in a 10×10 grid in another setting (Figure 5.4(b)). Each sensor node except the BS generates a packet randomly in each 10-second period. The packets are sent to the BS. We use some of the default settings in OPNET such as 1024-bit data packet and Geographic Routing Protocol (GRP) with a maximum of seven retransmissions before ACK is received. We set each node's initial trust value to be 0.99. We consider the cases of both single and multiple selective forwarding-based DoS attackers. The simulation time is set to be 30 minutes in the case of single attacker and 40 minutes for multiple attackers. We simulate the attacker(s) launch the proposed selective forwarding-based DoS attack to various numbers of victims. Both beta and entropy trust models (defined in equations (2.5) and (2.8)) as well as our enhanced trust mechanism (defined in equations (5.1) and (5.3)) with two avoidance strategies (CA and SA) are implemented in the OPNET Modeler for comparison purposes.

Table 5.3: OPNET simulation setup parameters

Parameters		Setting
General	Terrain dimension	2km × 2km
	Number of nodes	100
	Topology	Random/ Grid
	Max. simulation time	30 mins for single attacker; 40 mins for multiple attackers
	Base routing algorithm	GRP
	Max. retransmissions	7 (OPNET default)
Data packet Generation	Start time – Stop time	100 seconds – end of simulation
	Destination	Base station
	Packet arrival interval	Every 10 second
	Packet size	1,024 bits
Trust model	Type	Beta trust model and Entropy trust model
	Initial trust value	0.99
	Trust threshold (Θ_T)	0.7
Attack model	Number of attackers	Single attacker and Mulitple attackers (=2)
	Attack type	Selective forwarding-based DoS attack



(a) Random topology



(b) Grid topology

Figure 5.4: Two WSN topologies in our simulations. One hundred sensors are deployed in a 2km×2km area randomly in one setting (a) and in a 10×10 grid in another setting (b).

The main performance evaluation metrics are as follows:

1) Avoidance completion time (ACT): this is the time when all the victims have been re-routed to avoid the attacker.

2) False alarm rate (FAR): as discussed in the previous section, this is the probability that a good node is being considered as a selective forwarding-based DoS attacker.

3) Energy per packet (EPP): this is the average energy consumption to deliver a packet, regardless of whether the packet reaches the BS or not. EPP is obtained by the total energy consumed for data packet transmissions divided by the total number of data packets generated by all source nodes.

5.4.2 Simulation Results and Analysis of Single Attacker

ACT is the most important metric as it indicates the ability of each approach in identifying the attacker and re-routing the victim's packets. The simulation results on ACT in Table 5.4 reveal the followings:

1) Beta or entropy trust models alone fail to detect the attacker: In the grid topology, there are 21 source nodes that send packets to the monitoring node and then to the attacker. From Table 5.1, when the selective forwarding-based DoS attacker targets 6 victims or less, the beta trust model will not detect it; when it targets 3 victims or less, the entropy trust model cannot detect it. The results in Table 5.4 confirm this. This is also true for the random topology where the monitoring node forwards packets from 16 source nodes (including itself) to the attacker.

2) Our defensive mechanisms successfully detect the victims: Even when the attacker targets only one victim (the case of J=1), our defensive mechanism can help both the beta and entropy models to identify the victim node. The entropy model is quicker because a dropped packet will cause more reduction in the trust value in entropy model. It also takes a little more time for the SA strategy because it finds victims one by one.

3) Optimality of the proposed selective forwarding-based DoS attack: We already discussed in *1)* that our proposed selective forwarding-based DoS attack cannot be detected by the current trust model. Table 5.4 also shows that if the attacker becomes aggressive and targets more victims than the $V[j]$ values in Table 5.1 allow, then they will be detected by the existing approaches (both the beta model and the entropy model).

Table 5.4: Avoidance completion time *ACT* (in seconds): Top: Grid topology; Bottom: Random topology; CA: Complete Avoidance; SA: Selective Avoidance; J: Number of victim nodes.

GRID TOPOLOGY						
J	Beta trust model			Entropy trust model		
	Pure	CA	SA	Pure	CA	SA
1	Fail	542.5	542.5	Fail	269.0	269.0
2	Fail	539.0	551.5	Fail	266.0	279.5
3	Fail	538.5	552.0	Fail	268.0	278.5
4	Fail	538.0	553.5	255.5	255.5	255.5
5	Fail	543.0	562.5	184.5	184.5	184.5
6	Fail	541.5	552.5	152.5	152.5	152.5
7	581.0	541.5	561.0	151.5	151.5	151.5
RANDOM TOPOLOGY						
1	Fail	548.3	548.3	Fail	277.2	277.2
2	Fail	546.9	604.6	Fail	289.0	300.9
3	Fail	553.2	644.8	360.9	278.6	329.7
4	Fail	559.0	602.0	204.7	204.7	204.7
5	802.5	549.5	591.3	162.4	162.4	162.4
6	352.6	352.6	352.6	153.2	153.2	153.2

FAR measures the likelihood an approach will mistakenly treat an honest node as attacker. In the grid topology, there are very few collisions and there is no false alarm. The FAR values for different approaches in the random topology are shown in Table 5.5. This result confirms the claim of $\boldsymbol{FAR_{CA} \geq FAR \geq FAR_{SA}}$ we made in Theorem 5.2.

Table 5.5: False alarm rate *FAR* in the random topology. The result shows $FAR_{CA} \geq FAR \geq FAR_{SA.}$

J	Beta trust model			Entropy trust model		
	Pure	CA	SA	Pure	CA	SA
1	0.010	0.036	0.010	0.048	0.069	0.048
2	0.010	0.034	0.010	0.045	0.067	0.044
3	0.011	0.033	0.010	0.065	0.065	0.046
4	0.011	0.031	0.010	0.064	0.064	0.064
5	0.031	0.035	0.010	0.061	0.061	0.061
6	0.028	0.028	0.028	0.061	0.061	0.061

Finally, we report EPP. From Table 5.6, we can see that our proposed enhancement incurs very little energy overhead. In the avoidance approach where multiple paths are used, for a single attacker, two disjoint paths will guarantee the successful avoidance of the attacker. However, the energy consumption will be doubled. From energy perspective, our approach is much better than the current avoidance approach.

Table 5.6: Energy per packet *EPP* (*mJ*): Top: Grid topology; Bottom: Random topology; CA: Complete Avoidance; SA: Selective Avoidance; J: Number of victim nodes.

GRID TOPOLOGY						
J	Beta trust model			Entropy trust model		
	Pure	Overhead (%)		Pure	Overhead (%)	
		CA	SA		CA	SA
1	37.53	3.65	0.40	37.53	4.42	0.51
2	37.42	3.87	0.77	37.42	4.73	0.99
3	37.27	4.19	1.23	37.27	5.12	1.50
4	37.14	4.52	1.67	39.19	0	0
5	37.03	4.70	2.05	39.25	0	0
6	36.89	5.04	2.49	39.32	0	0
7	38.65	0.13	-2.07	39.3	0	0
RANDOM TOPOLOGY						
1	78.14	1.10	0.44	78.40	1.38	0.54
2	77.56	1.55	0.98	77.74	2.14	1.40
3	76.96	2.27	1.68	79.23	0.06	-0.67
4	76.42	3.01	2.30	79.55	0	0
5	78.08	0.54	0.09	79.70	0	0
6	79.01	0	0	79.51	0	0

In a couple of cases, when SA strategy is used, there is actually a small amount of energy savings. This is possible because the original geographical routing protocol does not guarantee energy efficiency. Moreover, as we have analyzed, SA strategy uses less energy than CA strategy because in SA strategy, only packets from detected victims will be re-routed.

5.4.3 Simulation of Multiple Attackers

For simplicity, we report the case of two attackers. When the two attackers are far away from each other, launching attacks to victim nodes independently, the result for each attack is almost identical to the single attacker case. Here we discuss the more interesting case when the two attackers are physically close to each other, for example, when the node to the right of the attacker in Figure 5.4(b) is also an attacker and they both target the same set of victims.

As one can imagine, when a victim node is identified, either the CA or the SA strategy will try to re-route packets to avoid the attacker. However, because the attacker's neighbor is also an attacker, if the monitoring node happens to choose the second attacker to forward packets to, both ACT and EPP will increase. In particular, the ACT will be around doubled because it will take about the same amount of time for the monitoring node to recognize the second attacker and re-route again. We now study the simulation results below:

First, we see that the two attackers together can target more victims without being detected. For example, in Table 5.4, we know that a single attacker will be detected by the pure beta trust model (without our defensive mechanism) if it attempts to attack 7 or more victims. However, Table 5.7 shows that the pure beta trust model can find the two attackers only when they are trying to attack 9 or more victims, which apparently indicates the improvement of attacking power.

Second, we see that the ACT is about tripled, instead of doubled, of the ACT in the single attack model. This is a little unexpected. However, the topology of the network and the position of the attackers are the main reason for this. In our case, when the

monitoring node finds the second attacker, it will re-route the packets to a new node. The new node happens to forward the packets to the second attacker again, thus it will take again time for the new node to identify the second attacker. This results in the ACT in the 2-adjacent attackers case is about three times of the ACT for single attacker.

Table 5.7: Avoidance completion time *ACT* (in seconds) in the case of multiple attackers in the grid topology

J	Beta trust model			Entropy trust model		
	Pure	CA	SA	Pure	CA	SA
1	Fail	1,896.6	1,895.3	Fail	805.3	807.3
2	Fail	1,876.6	1,447.3	Fail	780.6	632.0
3	Fail	1,862.6	1,306.6	Fail	772.6	574.6
4	Fail	1,861.3	1,230.0	8,750	742.6	624.6
5	Fail	1,869.3	1,189.3	617.3	563.3	475.3
6	Fail	1,866.6	1,158.0	372.0	372.0	372.0
7	Fail	1,864.0	1,137.3	298.0	298.0	298.0
8	Fail	1,374.0	999.3	248.6	248.6	248.6
9	1,438.3	1,092.6	835.3	215.3	215.3	215.3
10	740.0	740.0	740.0	196.6	196.6	196.6

5.5 Prevention Routing Algorithm

5.5.1 Motivation and Key Idea

As we discussed earlier, when an inside attacker relays packets for many sensor nodes in the network, it can pick one or more victims to launch the selective forwarding-based DoS attack. This is because it can hide its malicious behavior by forwarding packets from other nodes and maintaining a high trust value. If an attacker is on the routing path of only one or two nodes and it attacks a victim, the chance that the attacker will be detected quickly is high. In such a situation, the attacker may not take the risk to launch any attack. Based on this observation, we propose a prevention routing algorithm where an attacker has to choose between "not attacking" and "attacking and being caught". This is complementary to the detection and avoidance approach we described earlier. They can be used together as a more effective defensive mechanism.

The key idea of our prevention method is to limit the number of source nodes (N_{SMAX}) from which a node receives packets through the same monitoring node. As discussed in Section 5.2.2, if the attacker receives data packets from at least $V[0]$ source nodes from a monitoring node, it can launch the selective forwarding-based DoS attack against one of the source nodes without being detected by the monitoring node. Therefore, in our prevention method, we require that each monitoring node forwards packets from at most $V[0] - 1$ source nodes, that is $N_{SMAX} = V[0] - 1$. This will prevent the attacker from launching the selective forwarding-based DoS attack. If the attacker still launches the attack, it will be detected by the monitoring node.

Figure 5.5 shows how our prevention method successfully defends against a selective forwarding-based DoS attacker N1. Consider the beta trust model with the trust

threshold $\boldsymbol{\Theta_T}$ = 0.7 is used in a WSN. Node M receives packets from 8 nodes (nodes 1 to 8). Assume that N1 is the best neighbor of M and N2 is the next best choice of M to forward its packets to the BS. Although node N1 is M's best choice to relay the packets, our prevention routing algorithm will limit M to forward packets from only 3 nodes (nodes M, 6, and 7 in this case) to N1 because $V[0] = 4$ (see Table 5.1) and thus $N_{SMAX} = 3$. Packets from the other 6 nodes will be forwarded to nodes N2 and N3, three each.

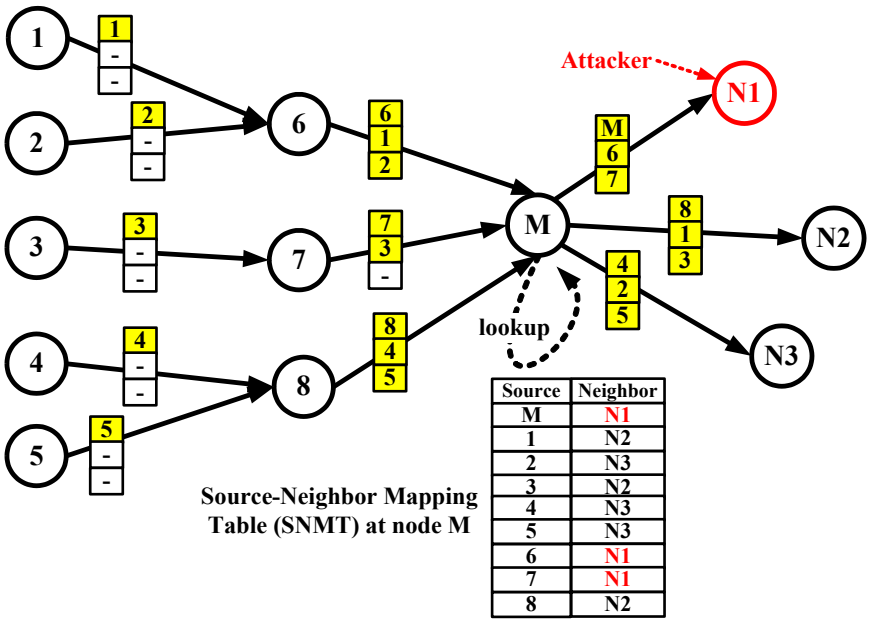


Figure 5.5: Our prevention routing algorithm against a selective forwarding-based DoS attacker (node N1) when the beta trust model is used with $\Theta_T = 0.7$.

The attacker N1 cannot launch the selective forwarding-based DoS attack against any of the three source nodes (M, 6, and 7) without being detected by M. If N1 starts attacking any of the three source nodes, N1's trust value evaluated by M will be 0.67 (=2/3) and thus N1 will be caught by M because N1's trust value is less than $\boldsymbol{\Theta_T}$(=0.7).

5.5.2 Proposed Prevention Routing Algorithm

Our prevention method can be easily integrated into any existing trust-based routing algorithm. Figure 5.6 shows the flow chart of a trust-based routing algorithm with our prevention method. Each time node M wants to forward a data packet toward the BS (regardless of its own packet or packets it receives from other nodes), M first checks the source node of the data packet and then finds a neighbor node A at M's *Source-Neighbor Mapping Table* (SNMT). SNMT is a lookup table that tells M which of M's neighbors will receive a certain source node's data packet to forward the packet toward the BS. If such node A is found at the SNMT for the source node, M will forward the packet to A. Otherwise, M will find a new neighbor node B such that the number of source nodes assigned to B is less than N_{SMAX}. If there are multiple neighbors satisfying such condition, the next hop selection algorithm of a base routing algorithm such as GPSR will choose the best one among them. If such node B is found, M registers B to its SNMT for the source node and then forwards the data packet to B. If M cannot find any neighbor satisfying such condition, M forwards the source's data packet back to the previous node P so that P can find other neighbor instead of M.

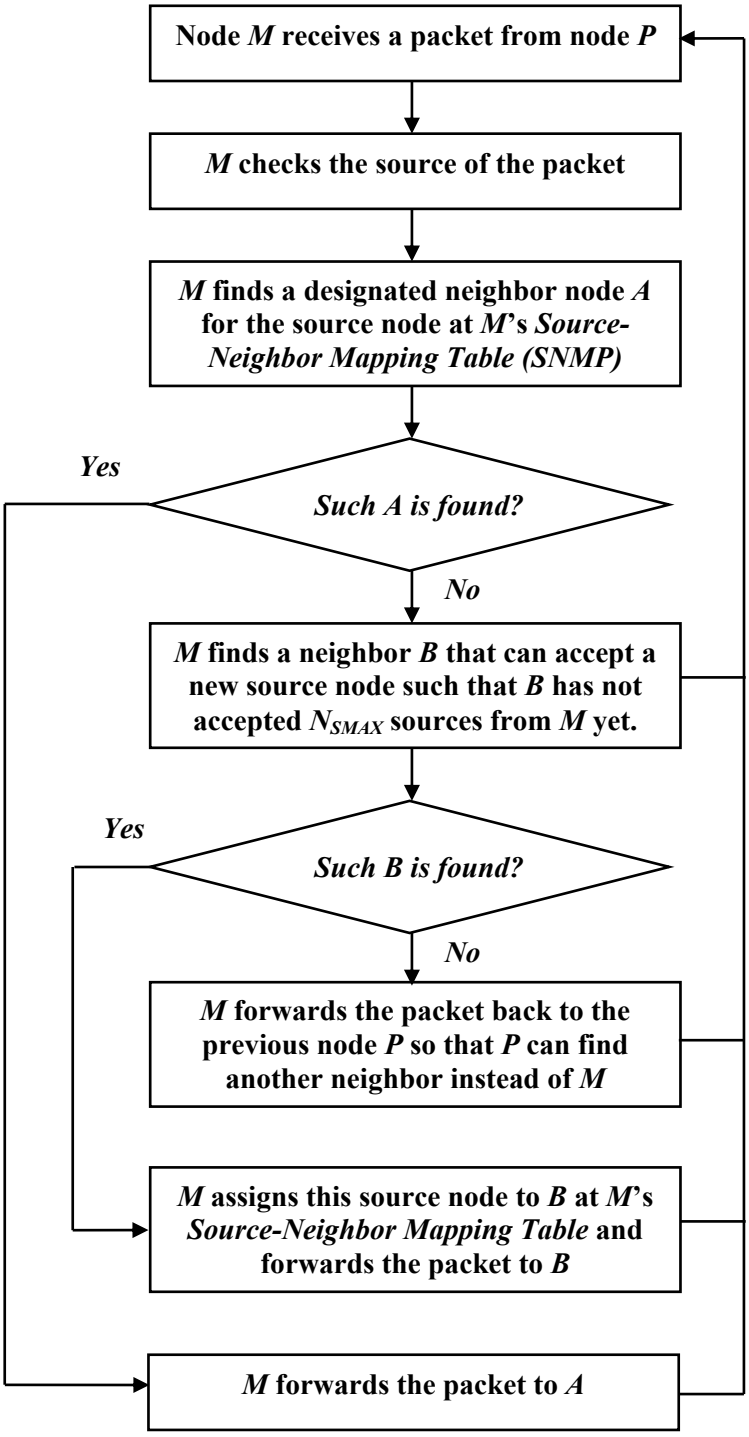


Figure 5.6: The flow chart of a trust-based routing algorithm with our prevention method to prevent the selective forwarding-based DoS attack.

We explain how a relay node M assigns source nodes to its neighbor nodes as shown in Figure 5.5. Assuming that every source's data packet is equally important, we use the FCFS (First Come First Serve) manner for this source-neighbor assignment process. For example, in Figure 5.5, assume that M received the first data packets of its eight source nodes in the following order: M, 6, 7, 8, 1, 3, 4, 2, and 5. Then, M assigns firstly arrived three source nodes (M, 6, 7) to its best neighbor N1 chosen by its base routing algorithm. The next three source nodes (8, 1, 3) and the remaining three source nodes (4, 2, 5) are assigned to M's next best neighbors N2 and N3, respectively. Each source-neighbor pairs is stored in M's SNMT. Whenever M receives a data packet, M will forward the packet to its designated neighbor associated with the source of the packet by using M's SNMT.

5.5.3 Simulation Setups and Results Analysis

We conduct simulations with the commercial network simulator OPNET Modeler v.17. One hundred sensors are deployed in a 2km×2km area randomly as shown in Figure 5.7. Each sensor node except the BS generates a packet randomly in each 10-second period. The packets are sent to the BS. We use some of the default settings in OPNET such as 1024-bit data packet and Geographic Routing Protocol (GRP) with a maximum of seven retransmissions before ACK is received. We set each node's initial trust value to be 0.99. We choose one sensor node near the BS as the selective forwarding DoS attacker so that it can receive and drop many data packets throughout the network. The simulation time is set to be 60 minutes (=3,600 seconds). The attacker targets various numbers of victims. We implement two trust-based routing algorithms: Trust-based GRP based on

the beta trust model (Beta GRP) and our prevention routing algorithm combining the Beta GRP and our prevention method (Beta GRP-P). For our prevention method, N_{SMAX} is set to be 3, because the beta trust model with $\Theta_T = 0.7$ is used in simulations.

Table 5.8: OPNET simulation setup parameters

Parameters		Setting
General	Terrain dimension	2km × 2km
	Number of nodes	100
	Topology	Random
	Max. simulation time	60 mins (3,600 seconds)
	Base routing algorithm	GRP
	Max. retransmissions	7 (OPNET default)
Data packet Generation	Start time – Stop time	100 seconds – end of simulation
	Destination	Base station
	Packet arrival interval	Every 10 second
	Packet size	1,024 bits
Trust model	Type	Beta trust model
	Initial trust value	0.99
	Trust threshold ($\boldsymbol{\Theta_T}$)	0.7
Attack model	Number of attackers	Single attacker
	Attack type	Selective forwarding-based DoS attack

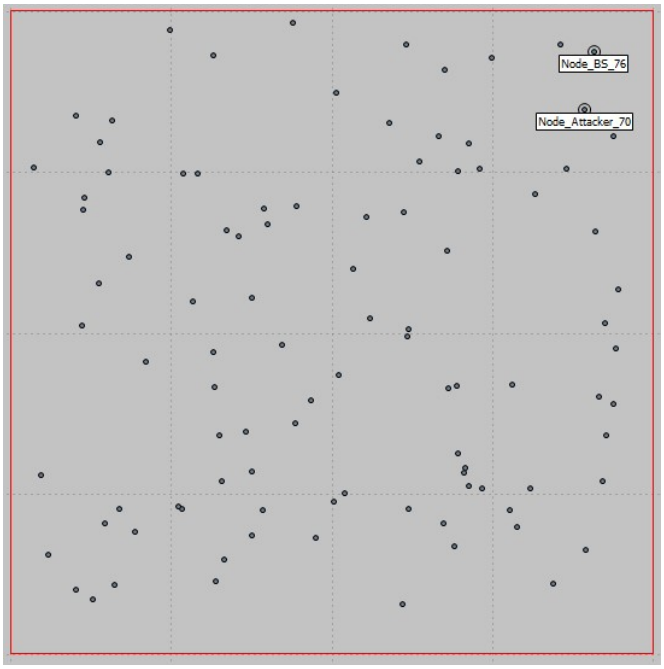


Figure 5.7: A WSN topology in our simulations. One hundred sensors are deployed in a 2km×2km area randomly

We use the following performance metrics:

1) Avoidance completion time (ACT): this is the time when all the victims have been re-routed to avoid the attacker.

2) False alarm rate (FAR): as discussed in the previous section, this is the probability that a good node is being considered as a selective forwarding-based DoS attacker.

3) Number of source nodes whose data packets route to the attacker through the same monitoring node (N_S): By this metric, we can get the theoretical maximum number of victims (N_{VMAX}) which the attacker can stealthily target without being noticed by the BS.

4) Packet delivery rate (PDR): This is the probability that a data packet is delivered to the BS. PDR is obtained by the total number of data packets delivered to the BS divided by the total number of data packets generated by all source nodes.

We first show how many source nodes' data packets can route through the inside attacker (located near the BS) in the simulation network topology. To see routing paths from source nodes to the BS via the attacker, we simulate the attacker forwarding packets normally toward the BS without attacking any source (attack off).

Figure 5.8 and Figure 5.9 show source nodes whose data packets route through the attacker and their routing paths to the BS when the Beta GRP and our prevention routing algorithm (Beta GRP-P) are used, respectively. We can see that when Beta GRP is used, the attacker receives data packets from many more source nodes as compared to our approach is used.

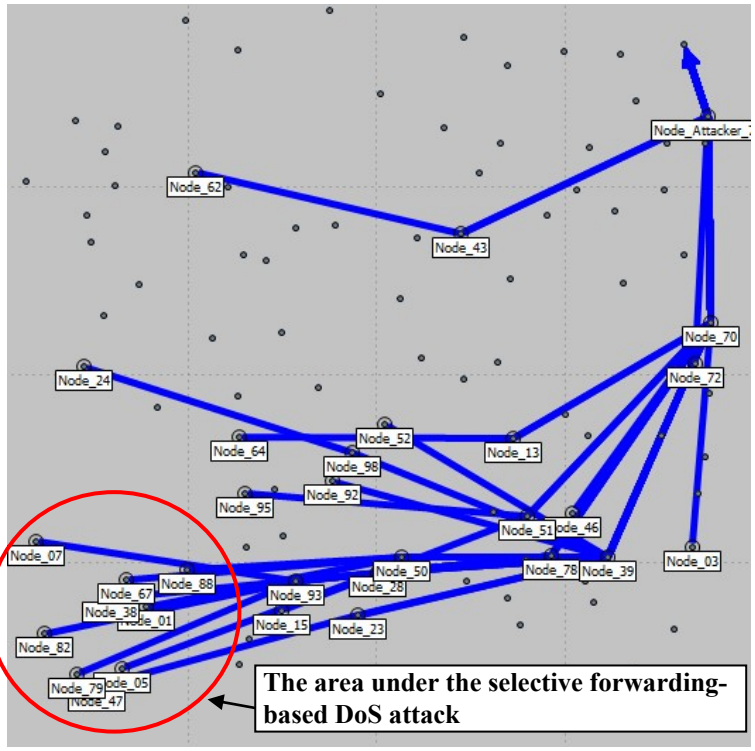


Figure 5.8: 30 potential victim source nodes and their routing paths to the BS when Beta GRP is used. The circle is the physical area under the selective forwarding-based DoS attack.

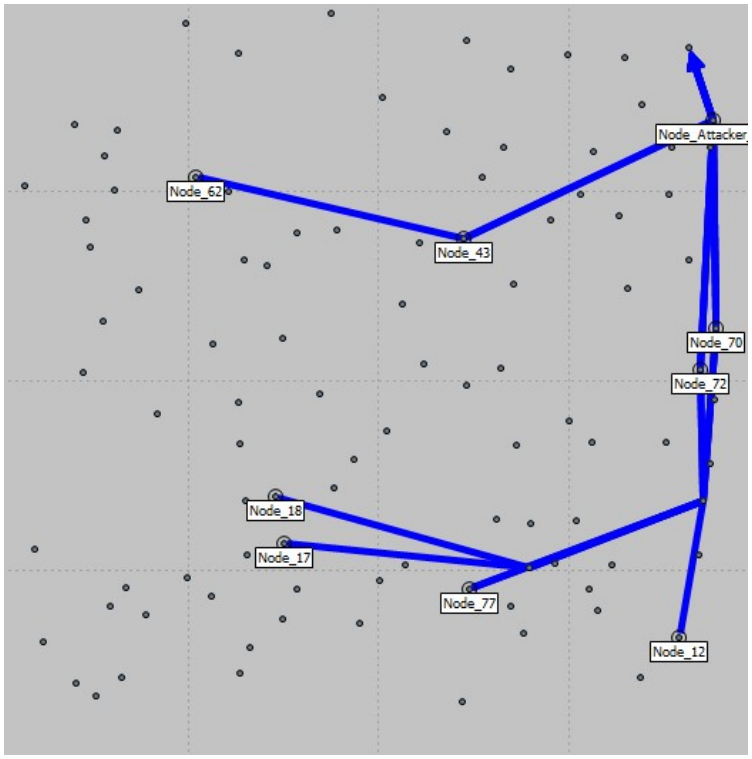


Figure 5.9: 8 potential victim source nodes and their routing paths to the BS when our prevention routing algorithm (Beta GRP-P) is used.

Table 5.9: The number of source nodes whose data packets route through the attacker (N_S) and the maximum number of victim source nodes (N_{VMAX})

Monitoring node	Beta GRP		Our Beta GRP-P	
	N_S	N_{VMAX}	N_S	N_{VMAX}
43	2	0	2	0
70	27	8	3	0
72	1	0	3	0
Total	30	8	8	0

Specifically, Table 5.9 shows N_S and N_{VMAX} of the Beta GRP and our approach (Beta GRP-P). When the Beta GRP is used, the attacker receives 27 source nodes' data packets from node 70. In this case, the attacker can drop up to 8 source nodes' data

packets completely without being detected by node 70's beta trust model theoretically (N_{VMAX}=8). Meanwhile, when our approach is used, the attacker receives at most 3 source nodes from node 70 or 72. Consequently, the attacker cannot successfully launch the DoS attack against any source node without being detected by node 70 or 72.

Second, we examine ACT that indicates the ability of each approach in identifying the attacker and re-routing the victim's packets. We simulate the attacker launching the selective forwarding-based DoS attack by increasing the number of victims (J). We assume that the attacker intentionally targets source nodes from node 70 because it can have the largest number of victim source nodes.

The simulation results on ACT in Table 5.10 reveal the followings:

1) Beta trust model alone fails to detect the attacker: As shown in Table 5.10, the attacker can attack up to 8 sources without being caught by node 70's beta trust model. As shown in Figure 5.8, the entire area monitored by the 8 victims (circled area) can be influenced by the DoS attack, and thus outside intruders can stay in or move around the area stealthily.

2) Our approach successfully defends against the attacker: The proposed outsider-insider colluding attack is not effective when our prevention approach is used because the number of victims is very small. That is, since the attacker cannot target more than 3 source nodes when our approach is used, the victim area is significantly reduced compared to when the Beta GRP is used. As a result, outside intruders' movement will be limited by the small area monitored by victim nodes. In addition, if the attacker insists to attack any victim (the case of J=1), the attacker will be detected by

1,296 seconds. This ACT can be reduced to around 540 seconds when our detection scheme is used together (see Table 5.4).

Table 5.10: Avoidance completion time *ACT* (in seconds): J: number of victim source nodes

J	Beta GRP	Our Beta GRP-P
1	Fail	1,296
2	Fail	360
3	Fail	216
4	Fail	
5	Fail	
6	Fail	
7	Fail	
8	Fail	
9	432	

Third, FAR measures the likelihood an approach will mistakenly treat an honest node as attacker. Both approaches have almost similar FARs that range from 0.03 to 0.04. Thus, we consider our approach does not increase FAR compared with the Beta GRP.

Finally, we report PDR. We show results when the number of victims (J) is less than 4 for comparison purposes. As shown in Table 5.11, our approach has a higher packet delivery performance than the Beta GRP. This is because our approach can detect and avoid the attacker while the Beta GRP cannot defend against the attacker.

Table 5.11: Packet delivery rate *PDR*: J: Number of victim source nodes

J	PDR	
	Beta GRP	Our Beta GRP-P
1	0.910	0.946
2	0.901	0.957
3	0.893	0.945

5.6 Summary

In this chapter, we first present a simple selective forwarding-based DoS attack, and show that two representative trust mechanisms (namely the beta trust model and the entropy trust model) fail to detect such attack. We also show the potential damage this attack could cause to the network. Second, we propose a source-level trust evaluation scheme to enhance the beta and entropy trust mechanisms to effectively detect the selective forwarding-based DoS attack. In addition, we propose two avoidance strategies to re-route the victim's packets so they can reach the BS, and validate our claims and evaluate the performance of our detection and avoidance mechanisms with extensive OPNET simulations.

Finally, we introduce a prevention routing algorithm to proactively prevent the selective forwarding-based DoS attack as a complementary defensive mechanism to our detection and avoidance methods. The preliminary simulation results show that the beta trust model with our prevention method successfully defends against a selective forwarding-based DoS attacker while the beta trust model fails in detecting the attacker.

There are also several directions for future work. First, how to further reduce ACT to minimize the attacker's damage to the network. Second, our preliminary results on network with lossy network show fairly large *FAR*. How to improve the accuracy of the proposed approach in such network is still a challenge. Finally, after the inside attackers become aware of our defensive mechanism, how they can respond to the challenge and launch more sophisticated attacks.

Chapter 6

Conclusion

In this dissertation, we analyze the limitation of the state-of-the-art trust mechanisms and propose several enhancement techniques to better defend against insider packet drop attacks in WSNs. The main contributions can be summarized as follows:

First, we demonstrate that the phenomenon of consecutive packet drops is one fundamental difference between attackers and good sensor nodes and build a hybrid trust model based on this observation to improve the detection speed and accuracy of current trust models. In Chapter 3, we design a lightweight trust model based on the concept of consecutive drops, and build a hybrid trust model that can adaptively choose between this new trust model and the existing beta trust model according to the number of consecutive drops. When a node is dropping packets consecutively, our new trust model will penalize the node based on the number of consecutive drops so the node's trust value will decrease fast. Extensive simulations on the OPNET Wireless Modeler show that our hybrid trust model outperforms the beta trust model in terms of important network performance parameters such as attack detection speed and attack detection accuracy. The results show that our hybrid trust model can always detect various inside packet drop attackers faster than the beta trust model. In addition, our hybrid trust model triggers false alarms on bad links due to collision, congestion, etc. and reduces natural packet drops by avoiding such links. A geographic routing protocol (GRP) with our hybrid trust model (Hybrid GRP) can deliver approximately 92% of data packets successfully even when 15% of the nodes in the network are grayhole attackers that

randomly drop 30% of packets. As a comparison, the pure GRP and GRP with the beta trust model (Beta GRP) can deliver only 71% and 78% of packets, respectively. Moreover, our Hybrid GRP consumes less energy to send a packet than the pure GRP and the Beta GRP.

Second, we devise a false alarm detection and recovery mechanism (FADER) that continues evaluating the trustworthiness of a node even when the node's trust value falls below the trust threshold (which will be considered as attacker by the current trust models) in order to determine whether the node is an attacker or a falsely detected good node. As described in Chapter 4, our main idea to treat false alarms in a trust-based routing is to give these nodes a second chance such that their neighbors can continue to monitor their behavior and evaluate their trustworthiness. By doing so, we can detect the false alarms and reuse the good nodes that have been mistakenly considered as attackers. This will enable us to improve the performance of the trust-based routing protocol in terms of many metrics such as the network lifetime, the packet delivery rate, and many routing performance measures. We have conducted extensive OPNET simulations and the results confirm these claimed advantages of our proposed FADER approach over a representative trust-based routing algorithm. The results show that FADER is able to recover 60−70% of the false alarms without mis-identifying any of the attackers as good nodes. In addition, with the false alarm detection and recovery feature, FADER can improve the network lifetime by 18.7−82.9% of Beta GRP and reduce the average hops per path by 7.5−16.7%.

Third, we demonstrate how intelligent inside packet drop attackers can successfully launch denial-of-service (DoS) attacks against many sensor nodes without

being detected by current trust models and propose effective detection and prevention methods against such attack, which we refer to as selective forwarding-based DoS attack. In Chapter 5, we first describe a simple selective forwarding-based DoS attack and show that the popular trust-based approaches (such as the beta trust model and the entropy trust model) for inside attacker detection fail to detect such attack. We also analyze the potential damage this attack can cause to the network. We then propose a source-level trust evaluation scheme to enhance the beta and entropy trust mechanisms for effective detection of the selective forwarding-based DoS attackers. Once the attacker is identified, we propose two avoidance strategies to re-route the victim's packets so they can be delivered successfully. As a complementary defensive mechanism to this detection and avoidance approach, we also introduce a prevention routing algorithm to proactively prevent the selective forwarding-based DoS attacks. The key idea of our prevention method is that each monitoring node will only forward packets from a limited number of source nodes to the next node. As a result, the next node will be forced to make a choice between "not attacking" or "attacking and being caught". In addition to a theoretical analysis of this approach, we also conduct extensive OPNET simulations to validate our claims and demonstrate the advantages of our proposed approaches.

Chapter 7

Future Work

Trust-based defense against insider packet drop attacks is a relatively new and very active research area. This dissertation contributes to this topic with several enhancements to existing trust-based mechanisms. Before we describe the immediate future work to improve and extend our research in Chapter 3 – 5, we discuss two major directions.

First, one of the characteristics that make insider packet drop attacks unique is the fact that inside attackers know the defensive mechanisms used in the network and therefore may adjust their attacking strategies accordingly. Although this is out of the scope of this dissertation, it will be interesting to study how intelligent inside attackers can break our proposed defending mechanism enhancements. The selective forward-based DoS attack we discussed in Chapter 5 is one example of insider-outsider colluding attacks. Another example is the case when the attacker does not have to drop a large number of consecutive packets to create the damage. In such scenario, an intelligent inside attacker can easily defeat our new trust model based on consecutive packet drops (Chapter 3). Also, an attacker can take advantage of the FADER approach in Chapter 4 and launch a mixed on-off and grayhole attack where the attacker will stop dropping packers when it becomes suspicious to the monitoring node and resume the attack when its trust value increases. This is the game that good guys and bad guys play in all security problems and it is this game that advances the arts of attacking schemes and defending

countermeasures (as we have discussed in Figure 1.1 in Chapter 1). Not coincidently, there are many recent game theoretical approaches to study this problem.

Second, for most of the study in this dissertation, we use the simple Beta trust model for simplicity to demonstrate how the current trust-based defending mechanisms can be improved with our proposed approaches and perform extensive simulation to validate our claims. Conceptually, the notion of *consecutive packet drops*, *false alarm recovery*, and *source-level monitoring* can all be integrated to other sophisticated trust models. As one can imagine, these models have better performance than the Beta trust model and therefore there will be less room for our enhancement techniques to gain further improvement. However, improving the detection accuracy, for instance, from 95% to 98%, is probably more challenging and more important (at least from the point of view of science) than improving it from 50% to 75%. It will be interesting to study some of the latest trust models and see how much our new techniques can help. Meanwhile, despite the fact that we do have several pieces of theoretical results, the lack of a mathematically sound foundation remains as a weakness for this dissertation. Empirical study has its value, but one theorem speaks more than one thousand experiments. It will be crucial to conduct theoretical study on the propose techniques to analyze their impact on the trust-based defensive mechanisms.

The next three paragraphs highlight some of the potential follow-up research for each of the enhancement techniques we have proposed in this dissertation.

In Chapter 3, we have introduced the concept of consecutive packet drops as a feature that can be used to distinguish packet drops by the inside attackers and caused by network problems. The intention is to argue that to further improve the detection speed

and accuracy of any trust-based mechanisms, it is necessary to include metrics (such as consecutive packet drops) that can differentiate attackers and good nodes. In the long term, we will continue to study the behaviors of inside attackers in order to develop such new metrics. Some immediate follow-up works of Chapter 3 include: (1) how to deal with the false alarms due to the large penalty to consecutive drops, particularly when the communication channel is lossy. (2) how to integrate the trust model on consecutive drops with models other than the Beta trust model, where the main difficulty is that there is no analytic solution for the model transition.

In Chapter 4, we have developed FADER to detect and recover the false alarms in trust-based defensive mechanisms. This work is motivated by the increasing amount of false alarms during our pursuit of fast and accurate inside attacker detection. We observe such phenomenon when we implement consecutive packet drop based trust model and we believe that this will happen for other approaches as well. The main goal of FADER (or false alarm recovery) is to accurately identify the false alarms and recover them as fast as possible. It is interesting that as a side effect, we are able to improve network lifetime and routing performance. The future work on this topic includes: (1) analyzing the impact of FADER on network lifetime and routing in order to obtain quantitative prediction on how much FADER can help. (2) investigating on how inside attackers may take advantage of FADER to disguise themselves as good nodes and providing countermeasures. (3) improving FADER's recovery ability (from the current 70% false alarm recover rate).

In Chapter 5, we have described an intelligent attack that can survive from the existing trust mechanisms and proposed new features to the trust-based method to defend such attack. This is an example showing the game between the attacker and defender,

where inside attackers explore the vulnerabilities of the trust methods and launch new attacks without being detected and defenders come up with new countermeasures to identify these new attacks. Future work along this direction includes: (1) how to further reduce the avoidance completion time in order to minimize the damage that the attackers can make to the network. (2) what other limitations in the current trust-based approaches and our enhancement techniques that can be utilized by the attackers to launch new attacks, particularly in the case when there are multiple collaborative attackers.

Finally, we will study how to build an integrated trust-based defense mechanism that combines all the defensive method enhancements proposed in this dissertation in the OPNET simulator. This simulation platform will be very useful for analyzing the intelligent attacker's behavior against our defense mechanisms and for validating the performance of any future follow-up works, especially in a large-scale WSN.

The big picture of our future work is shown in Figure 7.1.

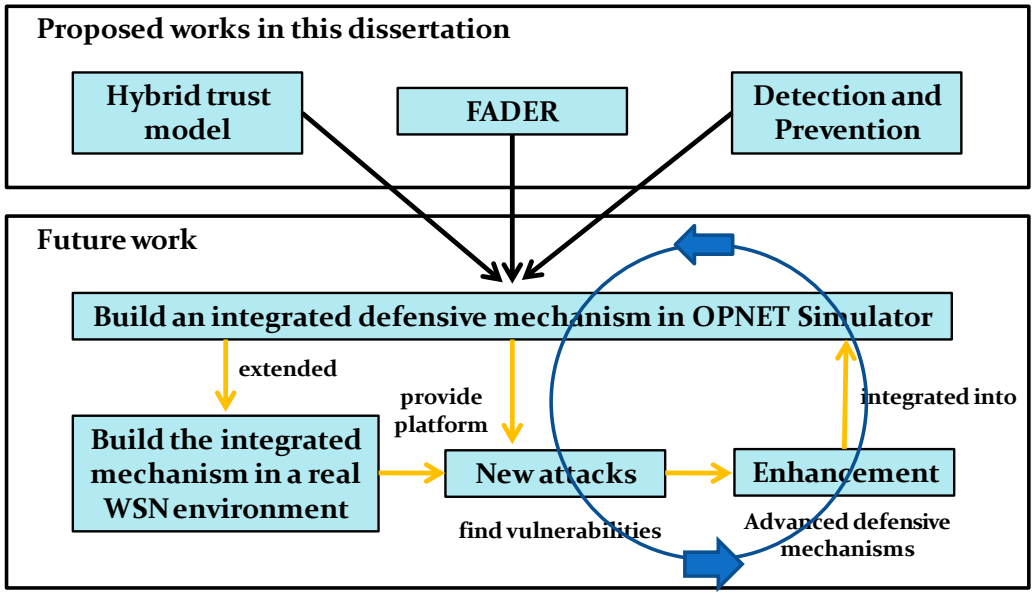


Figure 7.1: The big picture of our future work.

Appendix A

More In-depth Comparison between Hybrid GRP and Beta GRP

In this appendix, we report the performance of our hybrid trust model and the Beta trust model with various trust threshold values. This is in response to the following question raised by the committee members during the oral dissertation defense: *with the same trust threshold value, hybrid GRP has a much shorter detection completion time (DCT) than Beta GRP at the cost of a higher false alarm rate (FAR). If Beta GRP is allowed to create the same FAR by using a higher trust threshold value, what will be its DCT?*

First, we thank the committee members for this very insightful question and many other valuable comments. To answer this question, we have performed more simulation with various trust threshold values because it is hard to control FAR. More specifically, we use the following seven different threshold (TH) values: 0.60, 0.65, 0.70, 0.75, 0.80, 0.85, and 0.90 with the same OPNET simulation setups used in Figure 3.5 and Table 3.3. We deploy, near the base station, one grayhole attacker that drops 60% of packets (attack rate = 60%). For each threshold value, we run 10 simulations with 10 different random seeds, each simulation has a duration of three hours (10,800 seconds), and we report the average results of the 10 simulations.

We compare the performance of the two trust-based routing algorithms (Beta GRP and Hybrid GRP) along the following performance metrics:

1) Detection Completion Time (DCT): *DCT* is defined as the simulation time (seconds) when all attackers are detected by all the victim nodes whose packets are dropped by the attackers. *DCT* measures how quickly a trust model detects attackers.

2) Packet Delivery Rate (PDR): PDR is the probability that a data packet is successfully delivered to the base station (BS). A reliable trust-based routing will deliver most of the data packets to the BS even with the presence of inside packet drop attackers. We use PDR to evaluate the reliability of a trust-based routing algorithm.

3) False Alarm Rate (FAR): We adopt the definition of FAR from [23] where a false detection occurs when a node claims a non-attack as an attack. The *FAR* is the number of false detections divided by the number of normal nodes as defined in [23].

Table A.1: Performance metrics given various threshold values (attack rate = 60%)

Threshold TH		**0.60**	**0.65**	**0.70**	**0.75**	**0.80**	**0.85**	**0.90**
DCT	Beta	1998	1382.4	1058.4	777.6	496.8	324	205.2
	HYBD	378	334.8	280.8	259.2	226.8	151.2	118.8
N_A	Beta	547	387.7	280.3	200.7	139.7	92.7	55.8
	HYBD	95.1	86.6	70.1	60.2	49.4	37.9	29
N_{NA} (t = DCT)	Beta	848.7	617.8	474.1	352.2	231.4	154.3	108.9
	HYBD	538.8	422.5	330.8	250.8	171.5	120.4	84.7
PDR	Beta	0.926	0.935	0.946	0.955	0.961	0.970	0.977
	HYBD	0.959	0.960	0.963	0.966	0.971	0.976	0.980
FAR	Beta	0.106	0.114	0.125	0.129	0.154	0.208	0.225
	HYBD	0.127	0.129	0.145	0.156	0.195	0.222	0.231
$N_{T,\ BETA}$ (t = DCT)		9673.9	6712	5144.9	3800.7	2443	1612.5	1046.8
$N_{T,\ BETA}$ (t = 3 hours)		51404.4	51429.8	51434.2	51378.1	51394.3	51400.5	51394.8
$N_{NA,\ BETA}$ (t = 3 hours)		3218.2	2930.7	2468.4	2091.3	1825.2	1446	1090.4
NPDR (%)		5.68	5.17	4.31	3.66	3.26	2.59	1.95

The simulation results shown in Table A.1 and Figure A.1 reveal the followings:

First, as the threshold TH increases, DCT decreases because a trust model can detect the attacker quicker with a high TH. However, FAR increases because good nodes are more likely to be falsely detected as attackers when the trust threshold is high. In addition, PDR increases and becomes close to one because of the reduced DCT, the time when all the attackers are identified. After that, there will not be any packet drops due to attacking and network problems will be the only cause of packet drop. Our Hybrid GRP has higher FAR than Beta GRP because our Hybrid GRP considers consecutive drops and gives large penalty for them. However, for the same reason, our Hybrid GRP has smaller DCT and higher PDR than the Beta GRP.

Second, we provide answer to the committee's question. In order to get the same FAR, the beta trust model can use a higher threshold than our hybrid trust model. This improves DCT of the beta trust model, but it is still higher than our hybrid trust model. For example, when we set FAR = 0.129 (the red dotted horizontal line in Figure A.1), our hybrid trust model can use TH=0.65 and the beta trust model can afford to use TH = 0.75. When we draw the two red dotted vertical lines (in Figure A.1) from these two values, we see that the DCT of beta trust model (777.6s from Table A.1) is much higher than the DCT value (334.8s) of our model. This indicates that under FAR constraint, our hybrid model will still complete the attacker detection faster.

Third, it is not surprising to see that as the threshold TH increases, the number of packets dropped due to attacks (N_A) reduces because a trust model with a higher trust threshold can detect the attacker faster than a trust model with a lower trust threshold. In addition, as the threshold TH increases, the number of packets dropped due to non-

attacks (N_{NA}) decreases. This is because normal nodes in bad links (due to fading, congestion, etc) are detected as attackers by trust models since they drop many packets. Although these nodes are considered as false alarms in the insider attack detection problem, PDR can be improved by avoiding these problematic false alarm nodes, which indeed behave just like insider attacks.

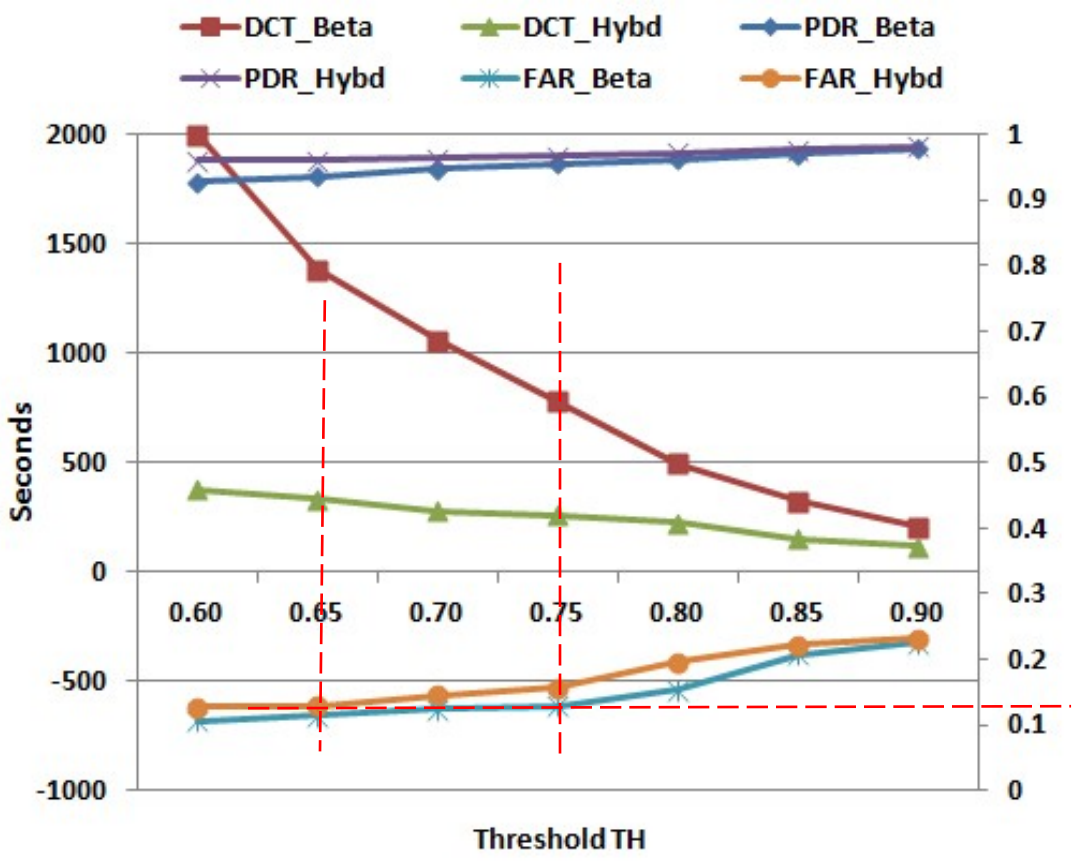


Figure A.1: *DCT* (left Y-axis), *PDR* and *FAR* (right Y-axis) for Hybrid GRP and Beta GRP with different trust threshold values (X-axis) when attacking rate is 60%.

Finally, we conduct further investigation in an attempt to explain how the OPNET simulator simulates natural packet drops caused by collision, fading, congestion, etc. after

we fail to obtain such information from the OPNET documentation. The last row in Table A.1 reports the natural packet drop rate (NPDR %), which is defined as the number of packet drops after DCT (these drops will be all natural packet drops as the attackers have been detected and packets are re-routed to avoid the attackers) divided by the number of packets generated after DCT. We calculate NPDR using the following formula

$$NPDR = \frac{N_{NA}(t = 3hours) - N_{NA}(t = DCT)}{N_T(t = 3hours) - N_T(t = DCT)} \times 100 \quad \text{(A.1)}$$

For example, when TH = 0.6, the average (over the 10 simulations) number of packets generated after DCT is 41,730.5 and the average number of packets dropped by non-attackers after DCT is 2369.5. Thus, NPDR is approximately 5.68% (=100×2369.5/41730.5). That is, 5.68% of packets generated are dropped naturally regardless of the attacker in this simulation setup in the OPNET simulator. As the threshold TH increases, NPDR decreases because good nodes with poor communication link will be misclassified by trust models as attackers after they drop many packets. Therefore, after DCT, they will not be included in the network and cannot cause more packet drops. In another word, with a high TH value, the network will experience less collision, fading, congestion, etc.

Bibliography

[1] J. -H. Cho, A. Swami, I. -R. Chen, "A Survey on Trust Management for Mobile Ad Hoc Networks," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 4, pp. 562–583, 2011.

[2] S. Djahel, F. Nait-abdesselam, and Z. Zhang, "Mitigating Packet Dropping problem in Mobile Ad Hoc Networks: Proposals and Challenges," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 4, pp. 658–672, 2011.

[3] G. Casella and R.L. Berger, "Statistical Inference," Duxbury Press, pp. 297–300, 1990.

[4] B. Yu and B. Xiao, "Detecting selective forwarding attacks in wireless sensor networks," *Proceedings of the 20th Parallel and Distributed Processing Symposium (IPDPS)*, April 2006.

[5] H.-M. Sun, C.-M. Che, and Y.-C. Hsiao, "An efficient countermeasure to the selective forwarding attack in wireless sensor network," *Proceedings of the IEEE Region 10 Conference (TENCON)*, pp.1–4, 2007.

[6] A. Josang and R. Ismail, "The Beta Reputation System," *Proceedings of the 15th Bled Electronic Commerce Conference*, June 2002.

[7] C. Karlof and D. Wagner, "Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures," *Ad Hoc Networks*, vol. 1, issue 2-3, pp. 293–315, 2003.

[8] B. Karp and H.T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (Mobicom)*, pp. 243–254, 2000.

[9] X. -S. Wang, Y.-Z. Zhan, S.-M. Xiong, and L.-M. Wang, "Lightweight defense scheme against selective forwarding attacks in wireless sensor networks," *Proceedings of International Conference on Cyber-Enabled Distributed and Knowledge Discovery (CyberC)*, pp. 226–232, 2009.

[10] S. Marti, T.J. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile and Ad Hoc Networks," *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (Mobicom)*, pp. 255–265, 2000.

[11] J. M. McCune, E. Shi, A. Perrig, and M.K. Reiter, "Detection of Denial-of-Service Attacks on Sensor Network Broadcasts," *Proceedings of IEEE Symposium on Security and Privacy (S&P)*, pp. 64–78, 2005.

[12] Y. L. Sun, W. Yu, Z. Han, and K. J. R. Liu, "Information Theoretic Framework of Trust Modeling and Evaluation for Ad Hoc Network," *IEEE Journal of Selected Areas in Communications*, vol. 24, no. 2, pp. 305–317, 2006.

[13] X. Liang and Y. Xiao, "Game Theory for Network Security," *IEEE Communications Surveys & Tutorials*, vol. 15, No. 1, pp. 472 – 486, First Quarter, 2013.

[14] A. D. Wood and J. A. Stankovic, "Denial of Service in Sensor Networks," *Computer*, vol. 35, issue 10, pp. 54–62, 2002.

[15] Y. Yu, K. Li, W. Zhou, and P. Li, "Trust Mechanisms in Wireless Sensor Networks: Attack Analysis and Countermeasures," *Journal of Network and Computer Applications*, vol.35, issue 3, pp. 867–880, 2012.

[16] T. Zahariadis, H.C. Leligou, P. Trakadas, and S. Voliotis, "Trust Management in Wireless Sensor Networks," *European Transactions on Tele-communications*, vol. 21, pp. 386–395, 2010.

[17] T. H. Hai and E.-N. Huh, "Detecting Selective Forwarding Attacks in Wireless Sensor Networks Using Two-hops Neighbor Knowledge," *Proceedings of Seventh International Symposium on Network Computing and Applications (NCA)*, pp. 325–331, July 2008.

[18] I. Khalil, S. Bagchi, C. N. Rotaru, and N. B. Shroff, "UnMask: Utilizing neighbor monitoring for attack mitigation in multihop wireless sensor networks," *Ad Hoc Networks*, vol. 8, issue 2, pp. 148–164, 2010.

[19] J. Zhang, W. Li, Z. Yin, S. Liu, X. Guo, "Forest Fire Detection System based on Wireless Sensor Network," *Proceedings of the 4th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pp. 520–523, 2009.

[20] J. Lopez, R. Roman, I. Agudo, and C. Fernandez-Gago, "Trust management systems for wireless sensor networks: Best practices," *Computer Communications*, vol. 33, no. 9, pp. 1086–1093, 2010.

[21] OPNET Modeler Wireless Suite, http://www.opnet.com/solutions/network_rd/modeler_wireless.html, 2012.

[22] A. A. Pirzada and C. McDonald, "Trusted Greedy Perimeter Stateless Routing," *Proceedings of the 15th International Conference on Networks*, pp. 206–211, 2007.

[23] Y. (L.) Sun, Z. Han, and K.J.R. Liu, "Defense of Trust Management Vulnerabilities in Distributed Networks," *IEEE Communications Magazine*, vol. 46, issue 2, pp. 112–119, 2008.

[24] Th. Arampatzis, J. Lygeros, and S. Manesis, "A Survey of Applications of Wireless Sensors and Wireless Sensor Networks," *Proceedings of the 13th Mediterranean Conference on Control and Automation*, pp. 719–724, 2005.

[25] T. Zahariadis, H. Leligou, P. Karkazis, P. Trakadas, I. Papaefstathiou, C. Vangelatos, L. Besson, "Design and Implementation of A Trust-aware Routing Protocol for Large WSNs," *International Journal of Network Security and Its Applications*, vol. 2, no. 3, pp. 52–68, 2010.

[26] W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," *Proceedings of the 33rd Hawaii International Conference on System Sciences*, pp. 3005–3014, 2000.

[27] M. A. Rassam, M. A. Maarof, and A. Zainal, "A Survey of Intrusion Detection Schemes in Wireless Sensor Networks," *American Journal of Applied Science*, vol.9, issue 10, pp.1636−1652, 2012.

[28] F. Bao, I.-R. Chen, M. Chang, and J.-H. Cho, "Hierarchical Trust Management for Wireless Sensor Networks and its Applications to Trust-Based Routing and Intrusion Detection," *IEEE Transactions on Network and Service Management*, vol. 9, no. 2, pp. 169−182, June 2012.

[29] S. Ganeriwal, L.K. Balzano, and M.B. Srivastava, "Reputation-based Framework for High-integrity Sensor Networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 4, issue 3, Article 15, May 2008.

[30] S. Ganeriwal and M. B. Srivastava, "Reputation-based framework for high integrity sensor networks," *Proceedings of ACM Security for Ad-hoc and Sensor Networks (SASN)*, pp.66−77, October 2004.

[31] I. Dietrich and K. Dressler, "On the Lifetime of Wireless Sensor Networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 5, issue 1, Article 5, February 2009.

[32] G. Zhou, T. He, S. Krishnamurthy, J. A. Stankovic, “Impact of radio irregularity on wireless sensor networks,” *Proceedings of International Conference on Mobile Systems, Applications, and Services (MobiSys)*, pp. 125−138, June 2004.

[33] Q. Yan, M. Li, T. Jiang, W. Lou, and Y. T. Hou, “Vulnerability and protection for distributed consensus-based spectrum sensing in cognitive radio networks,” *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, pp. 900−908, 2012.

[34] X. Su and R. V. Boppana, “On Mitigating In-band Wormhole Attacks in Mobile Ad Hoc Networks,“ *Proceedings of IEEE International Conference on Communications (ICC)*, pp. 1136−1141, 2007.

[35] Y. Cho, G. Qu, and Y. Wu, “Insider Threats against Trust Mechanism with Watchdog and Defending Approaches in Wireless Sensor Networks,” *Proceedings of IEEE Symposium on Security and Privacy Workshops (SPW)*, pp. 134−141, 2012.

[36] P. Kamat, Y. Zhang, W. Trappe, and C. Ozturk, “Enhancing Source-Location Privacy in Sensor Network Routing,” *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICSCS)*, pp. 599−608, 2005.

[37] Y. Li, J. Ren, and J. Wu, “Quantitative Measurement and Design of Source-Location Privacy Schemes for Wireless Sensor Networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 7, pp. 1302−1311, 2012.

[38] M. Johnson, M. Healy, P. V. D. Ven, M. J. Hayes, J. Nelson, T. Newe, and E. Lewis, “A comparative review of wireless sensor network mote technologies,” *Proceedings of IEEE Sensors*, pp. 1439−1442, 2009.

[39] J.-H. Chang and L. Tassiulas, “Energy Conserving Routing in Wireless Ad-hoc Networks,” *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, pp. 22−31, 2000.

[40] R. Madan, S. Gui, S. Lall, and A. Goldsmith, “Cross-Layer Design for Lifetime Maximization in Interference-Limited Wireless Sensor Networks”, *IEEE Transactions on Wireless Communications*, vol. 5, no.11, pp. 3142−3152, November 2006.

[41] A. Faridi, M. Rita Palattella, A. Lozano, M. Dohler, G. Boggia, A. Grieco, and P. Camarda, “Comprehensive Evaluation of the IEEE 802.15.4 MAC Layer Performance With Retransmissions,” *IEEE Transactions on Vehicular Technology*, vol.59, No.8, pp. 3917–3932, October 2010.

[42] K. Hoffman, D. Zage, and C. Nita-Rotaru, “A survey of Attack and Defense Techniques for Reputation Systems,” *ACM Computing Surveys*, vol 41, Issue 4, 2009.

[43] Leslie Lamport, Robert Shostak, and Marshall Pease, “The Byzantine Generals Problem,” *ACM Transactions on Programming Languages and Systems*, vol. 4, no. 3, pp. 382–401, July 1982.

[44] J. Li, R. Li, and J. Kato, “Future Trust Management Framework for Mobile Ad Hoc Networks,” *IEEE Communication Magazine*, vol 46, issue 4, pp.108–114, 2008.

[45] D. Trček, “Trust management in the pervasive computing era,” *IEEE Security & Privacy*, vol. 9, no. 4, pp. 52–55, July 2011.

[46] V. Varadharajan, “A Note on Trust-Enhanced Security,” *IEEE Security & Privacy*, vol. 7, Issue 3, pp. 57–59, May/June 2009.

[47] B. Xiao, B. Yu, C. Gao, "CHEMAS: Identify suspect nodes in selective forwarding attacks," *Journal of Parallel and Distributed Computing*, 67, pp. 1218–1230, 2007.

[48] Y. Challal, A. Ouadjaout, N. Lasla, M. Bagaa, A. Hadjidj, "Secure and efficient disjoint multipath construction for fault tolerant routing in wireless sensor networks," *Journal of Network and Computer Applications*, vol. 34, issue 4, pp. 1380−1397, 2011.

[49] David R. Raymond and Scott F. Midkiff, "Denial-of-Service in Wireless Sensor Networks: Attacks and Defenses," *Pervasive Computing*, vol. 7, issue 1, pp.74−80, 2008.

[50] S.-B. Lee and Y.-H. Choi, "A Resilient Packet-Forwarding Scheme against Maliciously Packet-Dropping Nodes in Sensor Networks," *Proceedings of the fourth ACM Workshop on Security of Ad Hoc and Sensor Networks (SANS)*, pp. 59−69, 2006.

[51] T. M. Cover and J. A. Thomas, "Elements of information theory," Wiley-Interscience, pp. 13−16, 2006.

[52] I. Demirkol, C. Ersoy, and F. Alagoz, "MAC Protocols for Wireless Sensor Networks: A Survey," *IEEE Communication Magazine*, vol. 44, issue 4, pp. 115–121, April 2006.

[53] R. Li, J. Li, P. Liu, and J. Kato, "A Novel Hybrid Trust Management Framework for MANETs," *IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW)*, pp. 251–256, June 2009.

[54] F. Zhao and L. Guibas, "Wireless sensor networks: an information processing approach," Elsevier, 2004.

[55] J. Gu, G. Qu, and Q. Zhou, "Information Hiding for Trusted System Design," *Proceedings of the 46th ACM/IEEE Design Automation Conference (DAC)*, pp. 698–701, July 2009.

[56] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "A Survey on Sensor Networks," *IEEE Communications Magazine*, vol. 40, no. 8, August 2002.

[57] C. Hartung, J. Balasalle, and R. Han, "Node Compromise in Sensor Networks: The Need for Secure Systems," *Technical Report CU-CS-990-05*, Department of Computer Science University of Colorado at Boulder, January 2005.

[58] B. Deb, S. Bhatnagar, and B. Nath, "ReInForM: Reliable Information Forwarding Using Multiple Paths in Sensor Networks," *Proceedings of IEEE Local Computer Networks (LCN 2003)*, pp. 406–415, October 2003.

[59] J. Deng, R. Han, and S. Mishra, "Intrusion Tolerance and Anti-Traffic Analysis Strategies for Wireless Sensor Networks," *Proceedings of IEEE International Conference on Dependable Systems and Networks (DSN)*, pp.594–603, 2004.

[60] W. Z. Khan, Y. Xiang, M. Y Aalsalem, and Q. Arshad, "Comprehensive Study of Selective Forwarding Attack in Wireless Sensor Networks," *International Journal of Computer Network and Information Security*, vol 3, no 1, pp. 1–10, 2011.

[61] L. Buttyan and J.-P. Hubaux, "Stimulating cooperation in self-organizing mobile ad hoc networks," *Journal of Mobile Networks and Applications*, vol. 8, issue 5, pp. 579-592, October 2003.

[62] S. Zhong, J. Chen, and Y. R. Yang, "Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks," *Proceedings of IEEE International*

Conference on Computer Communications (INFOCOM), pp.1987–1997, April 2003.

[63] Sophia Kaplantzis, Alistair Shilton, Nallasamy Mani, Y. Ahmet Sekercioglu, "Detecting Selective Forwarding Attacks in Wireless Sensor Networks using Support Vector Machines," *Proceedings of Intelligent Sensors, Sensor Networks and Information (ISSNIP)*, pp. 335 – 340, 2007.

[64] H.-d. Wang, Z. Yuan, and C.-d. Wang, "Intrusion Detection for Wireless Sensor Networks Based on Multi-agent and Refined Clustering," *Proceedings of Communications and Mobile Computing (CMC)*, pp.450 – 454, January 2009.

[65] M. Zarei, A. M. Rahmani, A. Sasan, and M. Teshnehlab, "Fuzzy based trust estimatioin for congestion control in wireless sensor networks," *Proceedings of International Conference on Intelligent Networking and Collaborative Systems (INCOS)*, pp. 233–236, November 2009.

[66] J. J. Jaramillo and R. Srikant, "DARWIN: Distributed and Adaptive Reputation mechanism for Wireless ad-hoc Networks," *Proceedings of the 13th Annual International Conference on Mobile Computing and Networking (MOBICOM)*, pp. 87–98, 2007.

[67] N. Meghanathan, "An Energy-aware Greedy Perimeter Stateless Routing Protocol for Mobile Ad hoc Networks," *International Journal of Computer Applications*, vol. 9, no. 6, pp. 30–35, November 2010.

[68] R. Haider, Dr. M. Y. Javed, N. S. Khattak, "EAGR: Energy Aware Greedy Routing in Sensor Networks," *Proceedings of Future Generation Communication and Networking (FGCN)*, pp. 344–349, 2007.

[69] Khalil and S. Bagchi, "Stealthy Attacks in Wireless Ad Hoc Networks: Detection and Countermeasure," *IEEE Transactions on Mobile Computing*, vol. 10, no. 8, pp. 1096–1112, August 2011.

[70] L. Moraru, P. Leone, and S. E. Nikoletseas, "Near optimal geographic routing with obstacle avoidance in wireless sensor networks by fast-converging trust-based algorithms," *Proceedings of the 3rd ACM Workshop on QoS and Security for Wireless and Mobile Networks (Q2SWinet '07)*, pp.31–38, 2007.

[71] S. Tanachaiwiwat, P. Dave, R. Bhindwale, and A. Helmy, "Location-centric Isolation of Mishbehavior and Trust Routing in Energy-constrained Sensor Networks," *Proceedings of IEEE International Conference on Performance, Computing, and Communcation (IPCCC '04)*, pp. 463–469, 2004.

[72] N. Bhalaji, S. Banerjee, and A. Shanmugam, "A Novel Routing Technique against Packet Dropping Attack in Adhoc Networks," *Journal of Computer Science*, vol. 4, issue 7, pp. 538–544, 2008.

[73] Poonam, K. Garg, and M. Misra, "Trust Based Multi Path DSR Protocol," *International Conference on Availability, Reliability and Security*, pp. 204 – 209, February 2010.

[74] A. A. Pirzada and C. McDonald, "Establishing trust in pure ad-hoc networks," Proceedings of the 27th Australasian Computer Science Conference (ACSC), pp. 47–54, 2004.