8-7-2008

Protecting 802.11-Based Wireless Networks From SCTS and JACK Attacks

Zhiguo Zhang
University of New Orleans

Protecting 802.11-Based Wireless Networks From SCTS and JACK Attacks

A Thesis

Submitted to the Graduate Faculty of the
University of New Orleans
in partial fulfillment of the
requirements for the degree of

Master of Science
in
Computer Science

by

Zhiguo Zhang

B.S. Harbin Institute of Technology University, 1997
M.S. Harbin Institute of Technology University, 1999

August, 2008

To my parents, my sisters, and my wife,
for their support as always.

Acknowledgements

I am grateful to all the people who made this work possible. Foremost, I would like to thank my advisor, Dr. Jing Deng, who was always very enthusiastic about the project and the results, whether positive or negative. He encouraged me to learn as much as possible and was always available when I needed his help or advice. Without his tolerance and support, this work not has come as far as it did. I really appreciate for the opportunities he gave me.

I would also like to thank my committee members, Dr. Shengru Tu and Dr. Adlai DePano, for their guidance and valuable comments on my thesis work.

In addition, I would like to acknowledge Jingqi Wu, a PhD student in Dr. Jing Deng's lab, for all of his input on this project. He was also very encouraging and supportive.

I would especially like to thank my wife, Xiaoyue, for supporting me throughout this entire process. Also, thanks to my parents for their unending support.

Contents

List of Figures

List of Tables

Abstract

The convenience of IEEE 802.11-based wireless access networks has led to widespread deployment. However, these applications are predicated on the assumption of availability and confidentiality. Error-prone wireless networks afford an attacker considerable flexibility to exploit the vulnerabilities of 802.11-based mechanism. Two of most famous misbehaviors are selfish and malicious attacks. In this thesis we investigate two attacks: Spurious CTS attack (SCTS) and Jamming ACK attack (JACK). In the SCTS, malicious nodes may send periodic Spurious CTS packets to force other nodes to update their NAV values and prevent them from using the channel. In the JACK, an attacker ruins legitimate ACK packets for the intention of disrupting the traffic flow and draining the battery energy of victim nodes quickly. Correspondingly, we propose solutions: termed Carrier Sensing based Discarding (CSD), and Extended Network Allocation Vector (ENAV) scheme. We further demonstrate the performance of our proposed schemes through analysis and NS2 simulations.

Keywords: IEEE 802.11 DC, Wireless networks security, Wireless LAN, Spurious CTS, Jamming ACK.

Chapter 1

Introduction

Since the world's first wireless local area network (WLAN), ALOHANET, emerged in 1971 at the University of Hawaii, the growth of the wireless network is significant [1]. Contrasted to the wired network, the wireless network is more flexible and convenient, especially for some situations where wired cable cannot reach, such as search/rescue after an earthquake, or communication in a battle field, and for those who prefer to mobile devices, such as laptop, Personal Digital Assistant (PDA) etc. There is no need to look for an Ethernet port when network connection is needed. People can get access to the network almost whenever and wherever they want.

Mobile Ad hoc Networks (MANETs) are autonomous systems of mobile nodes connected by wireless links. Each node operates not only as an end-system, but also as a router to forward packets [2]. The network topology is in general dynamic in nature. In MANETs the nodes operate in peer-to-peer fashion with no centralized base station, which makes the connectivity between the nodes quick and spontaneous.

In this thesis our research focused on the MANETs security issues: *selfish* and *malicious* attacks. We mainly investigate the following issues: Spurious CTS Attacks (SCTS), and Jamming ACK (JACK) attacks. The SCTS is a selfish attack, in which a misbehavior node will benefit to gain more opportunities to access the shared channel by prevent its neighboring nodes from transmitting. The JACK is a malicious attack, in which an adversary consumes a small mount of energy to jam the medium and drain the energy of the victim nodes as quickly as possible.

1.1 Background

In the following section, we will introduce some fundamental concepts before we present details of our investigations based on wireless networks security issues.

1.1.1 Wireless Local Area Networks (WLANs)

The major inconvenience of LANs is that the physical link restricts its applications, especially in cases of emergence or battle situations. To avoid the wiring associated with the interconnection of PCs in LANs, researchers have explored the possible usage of radio waves and infrared light for interconnection [1]. This has resulted in the emergence of WLANs. Besides of the main difference between wired and wireless networks: transmission medium, other technical differences are listed bellow:

1. **Address is not equivalent to physical location**: in a wireless network, address refers to a particular station need not be stationary. Therefore, address may not always refer to a particular geographical location.
2. **Dynamic topology and restricted connectivity**: the mobile nodes may often go out of reach of each other. This means that network connectivity can be partial at times.
3. **Medium boundaries are not well-defined**: the exact reach of wireless signals cannot be determined accurately. It depends on various factors such as signal strength and noise levels.
4. **Error-prone medium**: transmissions by a node in the wireless channel are affected by simultaneous transmissions by neighboring nodes that are located within the direct transmission range. This means that the error rates are significantly higher than that in wired cables. Typical bit error rates are of the order of 10^{-4} in a wireless channel as against 10^{-9} in fiber optic cables [1].

1.1.2 Mobile Ad Hoc Networks (MMANETs)

The performances of mobile ad hoc networks (MANETs) have increased tremendously in the last few years. Comparing with wired networks, the unique characteristics of MANETs, which are open network architecture, shared wireless medium, stringent resource constraints, and highly dynamic network topology, present a new set of challenges to security design [3]. Unfortunately, existing solutions in traditional wired networks are not suitable to tackle MANETs' vulnerabilities [4].

The advantages of MANETs bring along some disadvantages. Some critical attributes of MANETs on which we focused in this thesis are listed bellow.

1. **Transmission/receiving range**: as the name of WLAN implies, the medium of communication between wireless nodes is radio wave in the air. Such wireless connections are unavailable when nodes are outside the transmission range of pre-existed base stations. And because of some restrictions, such as physical size, tethered power, and commercial factor, mobile nodes' transmission range cannot rival that of wired produces.
2. **Carrier sensing range**: same to LANs, a node in MANETs cannot receive two messages coming from two directions simultaneously (that is data collision, just like only one voice is permitted at any time in a telephone conference). Because in wired networks all nodes are linked with cables, each node can detect others' status easily and directly before it gets permission to transmit data. Unfortunately, in MANETs, the limit of carrier sensing range restricts the capability of detecting other nodes' status. Complicated protocols adopted to solve that inherent attributes of WLANs will be discussed later.
3. **Security**: a MANET is a collection of mobile nodes, which can communicate each other directly (when these are in its transmission range) or relay on nodes as routers (when these are out of its transmission range). Processing as a person in a human society, an individual mobile node of a MANET may attempt to benefit from other nodes, but refuse to share its own resources. Such nodes are called selfish or misbehaving nodes, and their behavior is termed selfishness or misbehavior [5].
4. **Energy**: because of the bottleneck of battery technology and economical consideration, the nondurable battery is one of the most critical restrictions of the development of MANETs. One of the major sources of energy consumption in mobile nodes of MANETs is wireless transmission [3, 6]. A selfish node may refuse to forward data packets for other nodes in order to conserve its own energy, or cheat to work as a router without forwarding.

The randomness of protocol operation together with the inherent difficulty of monitoring in the open and highly volatile wireless medium poses significant challenges comparing with wired networks, WLAN's protocols are complicated and inefficient [7]. To make things worse, that makes WLANs susceptible to sophisticated MAC layer Denial of Service (DoS) attacks [8].

1.1.3 Hidden Nodes

Because of the specializations of open media and range limitation, wireless linked nodes sharing the same channel cannot recognize the status of the others who reside outside their sensing range. Data collision cause by exposed/hidden terminal problems is the main vulnerability of wireless networks comparing with wired networks [9]. Many Multiple Access Control MAC schemes have been designed to solve these problems. In particular, the IEEE 802.11 DCF MAC scheme employs the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) technique.

The hidden terminal problem refers to the collision of packets at a receiving node due to the simultaneous transmission of those nodes that are not within the direct transmission range of the sender, but are within the transmission range of the receiver. Collision occurs when both nodes transmit packets at the same time without knowing about the transmission of each other. In Figure 1.1, if both node A and C transmit to node B at the same time, their packets collide at node B.

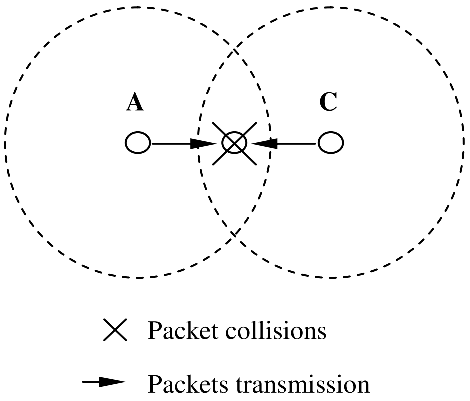

Figure 1.1 The hidden node problem

1.1.4 Exposed Nodes

The exposed terminal problem refers to the inability of a node, which is blocked due to transmission by a nearby transmitting node, to transmit to another node. Consider the example in Figure 1.2. Here, if a transmission from node C to node D is already in progress, node B cannot transmit to node A, as it concludes that its neighbor node C is I transmitting mode and hence it should not interfere with the on-gonging transmission.

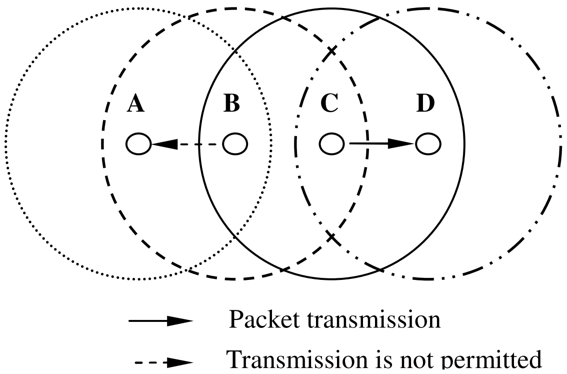

Figure 1.2 The exposed node problem

1.1.5 Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA)

Carrier sense with multiple access and collision avoidance is the MAC layer mechanism used by IEEE 802.11 WLANs. Carrier sense with multiple access and collision detection (CSMA/CD) is a well-studied technique in IEEE 802.x wired LANs. This technique cannot be used in the context of WLANs effectively because the error rate in WLANs is much higher and allowing collision will lead to a drastic reduction in throughput [1]. Moreover, detecting collisions in the wireless medium is not always possible. The technique adopted here is one of collision avoidance.

The basic channel access mechanism of IEEE 802.11 is shown in Figure 1.3 [10]. If the medium is sensed to be idle for a duration of DIFS, the node accesses the medium for transmission. Thus the channel access delay at very light loads is equal to the DIFS. If the medium is busy, the node backs off, in which the station defers channel access by a random amount of time chosen within a contention window (CW). As soon as the back-off counter reaches zero and expires, the station can access the medium. During the back-off process, if a node detects a busy channel, it freezes the back-off counter and the process is resumed once the channel becomes idle for a period of DIFS. Each station executes the back-off procedure at least one between every successive transmission [1].

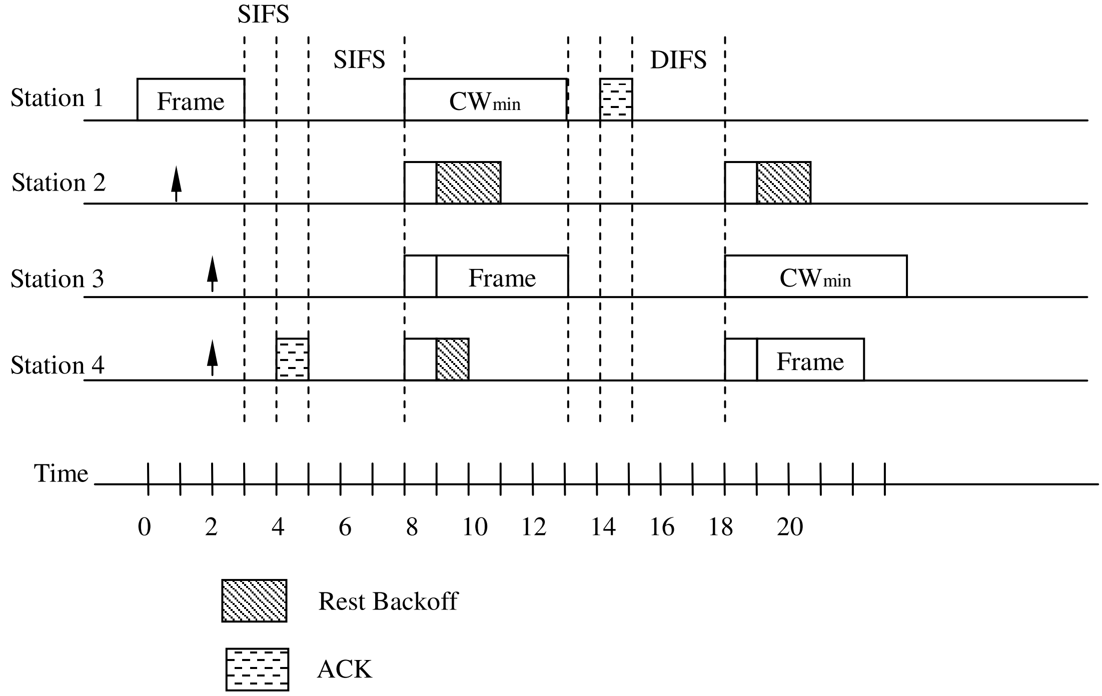

Figure 1.3 IEEE 802.11 DCF channel access mechanism

In the Figure 1.3 each event is illustrated as following:

T=1 Station 2 wants to transmit but the media is busy.

T=2 Stations 2 and 4 want to transmit but the media is busy.

T=3 Station 1 finished transmission.

T=4 Station 1 receives ACK from its transmission (SIFS = 1).

T=5 Medium becomes free

T=8 DIFS expires. Station 2, 3, 4 draw backoff count between 0 and 5.

The counts are 3, 1, 2.

T=9 Station 3 starts transmitting.

Station 2 and 4 pause backoff counter at 3 and 1 resp.

T=13 Station 3 finishes transmission.

T=14 Station 3 receives ACK.

T=15 Medium becomes free.

T=18 DIFS expires

Stations 2, 4 start their backoff counter.

T=19 Station starts transmitting.

The IEEE 802.11 DCF MAC scheme employs the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) technique [11], in which the so-called Request-To-Send and Clear-To-Send (RTS/CTS) exchange is used.

The RTS/CTS exchange works as follows: A client wishing to transmit a message sends an RTS packet, which includes source address, destination address, and duration for the transmission. The receiver responds with a CTS packet if the channel is free, with duration information in it. After receiving the CTS packet, the sender responds with data packets and the receiver sends an acknowledgment to inform the sender that the transmission has completed. All the exposed nodes (which are within the transmission range of the sender but out of that of the receiver) overhearing the RTS packet and all the hidden nodes (which are within the transmission range of the receiver but out of that of the sender) overhearing the CTS packet keep silent during the transmission duration which guarantees the successful transmission and reception of the message (cf. Figure 1.4).

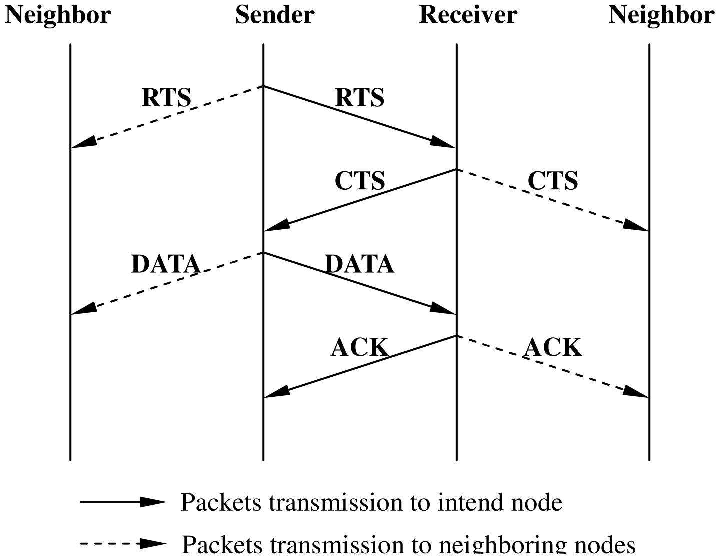
Neighbor
Sender
Receiver
Neighbor
RTS
RTS
CTS
CTS
DATA
DATA
ACK
ACK
Packets transmission to intend node
Packets transmission to neighboring nodes

Figure 1.4 Packet transmission in CSMA/CA

1.1.6 Network Allocation Vector (NAV)

In IEEE 802.11 DCF, a Network Allocation Vector (NAV) is implemented for channel reservation. The NAV is a timer that indicates the duration for which the medium has been reserved. The sender sets the Duration/ID field of its RTS packet equal to the time for which it expects to use the medium, including the transmission time of all the packets in the sequence. Every exposed node updates its NAV accordingly after overhearing the RTS packet (the NAV value is changed only when the new value is greater than the current NAV). CTS packets have the same field to be used by overhearing nodes in a similar manner.

Nodes set up a timer to count down the NAV. When the NAV is greater than zero, the so-called virtual carrier-sense function indicates that the medium is busy. Nodes can only start transmission when both the physical carrier-sense function and the virtual carrier-sense function indicate an idle medium. So in this way, the medium is reserved for a sender/receiver pair until the end of the transmission.

1.2 Spurious CTS Attack (SCTS)

The IEEE 802.11 DCF MAC providing the NAV as a virtual carrier-sense function for channel reservation can effectively mitigate data collisions in MANETs. However, there is no mechanism to validate the NAV values of RTS/CTS packets. When a node overhears an RTS or CTS packet, it does not know whether the corresponding NAV value is legitimate or not. This makes spurious packet transmission an attractive approach for malicious nodes to disrupt communications. For example, if a malicious node sends a spurious CTS packet, in which it intentionally puts in a long NAV timer, other nodes within the transmission range will set up their NAVs equal to this value without suspicion. These nodes will wait to access the channel for the entire NAV period while the channel is idle. This vulnerability may be exploited by attackers to block neighboring nodes from accessing the shared medium for an extended period of time. Such attacks reduce the throughput especially when the node density is high.

In Chapter 3 of this thesis, we investigate the effect of SCTS attacks and propose a solution to address this problem. Our solution is termed Carrier Sensing based Discarding mechanism (CSD). The main idea of the CSD scheme is to ask nodes overhearing CTS packets to look for the expected data packet transmission. If no transmission carrier is sensed, the CTS packet is treated as spurious and the NAV value on the CTS packet is discarded. Such detections of carrier on the channel may be performed multiple times before discarding the NAV in order to overcome potential missed detections.

1.3 Jamming ACK attack (JACK)

To helping us understand the JACK attack, we can imagine a scene: there is a border between two opposing countries, battle field. It appears that is an area without physical fighting, but actually it permeates drastic conflict between sensors. Trying to eavesdrop the opponent movement, country A deployed plants of wireless sensors on the border to monitor noise, infrared radiation, and vibration. But soon, he finds messages transmit in the network are delayed seriously, and some sensors’ battery are run out quickly. Through investigation, they realize the sensors suffer from a special attack: JACK attack.

Medium Access Control (MAC) schemes are designed to reduce such collisions and to improve the channel usage efficiency. In most of these MAC schemes, an Acknowledgment (ACK) packet will be transmitted from the data receiver to the data sender after the data packet is successfully received. An example is the widely implemented IEEE 802.11 DCF that employs CSMA/CA (we will show that most discussions apply to other MAC schemes with ACK packets). In CSMA/CA, each pair of hosts will go through the process of Request-To-Send packet, Clear-To-Send packet, Data packet, and ACK packet (RTS/CTS/DATA/ACK) to reserve and to use the medium exclusively. Besides using physical carrier sensing, the CSMA/CA scheme employs a virtual carrier sensing technique with the help of NAV. Neighbors overhearing the NAV information are required to keep silent until the NAV expires.

Many hosts in wireless networks such as MANETs are usually powered by batteries. How to make good use of the limited energy is always one of the top concerns. In some applications, such as battlefield, hosts in MANETs will face many adversaries. Besides jamming the medium and preventing targeted hosts from using it, attackers may also try to drain the energy of the victim nodes as quickly as possible.

In Chapter 4 of this thesis, we investigate a potential attack, Jamming ACK (JACK) attack, to wireless networks. JACK attackers basically send out packets to collide with other legitimate ACK packets so that the data sender will need to reschedule the data transmission. Therefore, JACK attack can be used by adversaries to drain energy level of victim nodes and this is achieved with a small amount of energy. In order to mitigate the effect of the JACK attacks, we propose an effective countermeasure called Extended NAV (ENAV). The basic idea of ENAV is to extend the ACK packet transmission window so that it becomes more difficult for the JACK attackers to jam the legitimate ACK packets.

In this thesis, we investigate the damaging effect of the JACK attacks to wireless networks and the effectiveness of our ENAV scheme protecting MAC schemes from such attacks. We will also derive and evaluate the best length of the NAV extension in the ENAV scheme in this thesis.

1.4 Thesis Organization

The remainder of this thesis is structured as follows:

In Chapter 2, we introduce the related works. Some approaches, referring to selfish and malicious nodes' trying to access unfair share of channel denying the neighboring nodes access to the channel, and solutions to counter them are listed. In Chapter 3, we explain the details of the SCTS attack and the CSD mechanism as a solution. In Chapter 4, we investigate JACK and its solution ENAV. In Chapter 5, we present our concluding remarks.

Chapter 2

Related works

Most research related to misbehaving nodes in the wireless network addresses selfish and malicious misbehavior. Selfish misbehavior implies that the selfish nodes misbehave with the intention to improve its own performance in terms of throughput, latency, energy etc. Malicious misbehavior intends to disrupt normal network operation with no performance gain to the misbehaving node. The security problem and the misbehavior problem of wireless networks including MANETs have been studied by many researchers, e.g., [14, 15, 16, 17]and [18].

In MANETs, nodes can organize themselves in a network without any help of a predefined infrastructure. To cooperate properly, each node should strictly follow the rules defined by standard routing protocols, medium access control protocols, etc. For individual advantage nodes might not cooperate though. S. Buchegger et al. [16] presented a hybrid scheme of selective altruism and utilitarianism to resist selfish behavior nodes.

During Spurious RTS/CTS attacks, malicious nodes access an unfair share of the channel by manipulating the duration value in their control packets, i.e., setting the NAV falsely high. In IEEE 802.11, to reduce the risk of Denial-of-Service (DoS) via the use of fake RTS packets, a node is permitted to reset its NAV if no PHY-RXSTART [19] indication is detected from the Physical layer (PHY) some time after receiving the RTS packet.

Parker et al. [20] simulated such an RTS attack and proposed a scheme to accurately diagnose malicious attacks in ad hoc networks by combining the input from all layers of the network stack. Acharya et al. [21] investigated fake RTS attacks in IEEE 802.11b networks, using a single fake RTS jammer and proposed the CTSR (CTS Reservation) protocol based on assessment of the channel status and resetting NAV value if the channel is idle. These papers made an attempt to detect the attacks caused by Spurious RTS packets, but our paper deals with the attacks caused by Spurious CTS packets.

Chen et al. [22] investigated Spurious RTS/CTS attacks and NAV attacks. A solution for NAV attacks was proposed and a protocol modification was recommended that IEEE 802.11

should have a provision to reset the NAV value after a fixed period of time if the channel is found idle.

Takai et al. [23] proposed that it is necessary to determine ad hoc wireless network performance with the consideration of the physical layer. Their illustrated that slight inaccuracy at physical layer can magnify inaccuracy at the higher layer protocols. The set of factors at the physical layer such as signal reception, path loss, fading, interference and noise computation, and preamble length are relevant to the performance evaluations of higher layer protocols. Their research includes studying the impact and comparison of the above mentioned factors through two commonly used simulators NS-2 and GloMoSim.

Bellardo et al. [24] dealt with the DoS in the 802.11 MAC protocol. They focused on the threats posed by denial-of-service (DoS) attacks against 802.11's MAC protocol. Such attacks, which prevent legitimate users from accessing the network, are a vexing problem in all networks, but they are particularly threatening in the wireless context. Without a physical infrastructure, an attacker is afforded considerable flexibility to decide where and when to attack, as well as enhanced anonymity due to the difficulty in locating the source of individual wireless transmissions. Moreover, the relative immaturity of 802.11-based network management tools makes it unlikely to diagnose a well-planned attack quickly [25, 26]. They described implemented and evaluated non-cryptographic countermeasures [27] that can be implemented in the firmware of existing MAC hardware.

Ray et al. [28] investigated the channel blocking problem and proposed a solution for RTS induced Congestion due to virtual blocking. They used an RTS validation technique to solve the so-called "false blocking" problem. Through assessment the technique checks the status of medium and reset the virtual carrier sense indicator, NAV, if the channel is idle.

Kyasanur and Vaidya [29] investigated the misbehavior of selfish nodes that intentionally disobey the MAC protocol rules in IEEE 802.11 networks. These misbehaving hosts may wait for smaller back-off intervals to gain unfair share of the channel compared to other honest hosts. A protection scheme was proposed to detect and penalize any selfish misbehavior. In this scheme, the receiver selects a random back-off value and sends it in the CTS and ACK packets to the sender. The sender must use this assigned back-off value in its next transmission to the receiver. A receiver observes the back-off time between consecutive transmissions from the same sender and judges whether the sender is deviating from the protocol. The application of the proposed scheme in networks with more than one receiver is however more difficult. Also Cardenas et al. [30] focus on the prevention and detection of the manipulation of the back off mechanism by selfish nodes in 802.11 networks. They proposed a detection algorithm to ensure honest back off when at least one, either the receiver or the sender is honest.

Like most other protocols, CSMA/CA was designed with the assumption that the nodes would play by the rules. However, we claim that this assumption is less and less appropriate, because the network adapters are becoming more and more programmable [45, 46]. Cagalj et al. [32] used a game-theoretic approach to investigate the problem of the selfish behavior of nodes, which break the rules to obtain a much larger share of the available bandwidth at the expense of other users.

Virtual carrier-sense is used to determine the availability of the shared medium in the IEEE 802.11 MAC protocol. Chen et al. [22] investigated the vulnerabilities that a misbehaving node may exploit to block neighboring nodes from accessing medium for an extended period of time. Two potential virtual jamming attacks were discovered. A backward-compatible solution, NAV Validation, was designed to overcome these vulnerabilities. The main idea of NAV Validation is to set two MAC-layer timers: one timer monitors the duration between RTS packet and DATA packet; the other monitors the duration between CTS packet and ACK packet. The two timers help to double-check whether the DATA and ACK packets appear as expected. Besides the virtual jamming attacks, the hosts may also suffer from physical jamming attacks such as the Jamming ACK attack that we will discuss in this thesis.

Xu et al. [33] investigated DoS attacks at MAC layer in wireless networks. Four types of attacks were categorized: constant jammer, deceptive jammer, random jammer, and reactive jammer. Constant jammer continually emits a radio signal. It is effective, but it costs too much energy and is easy to detect. Deceptive jammer constantly injects regular packets without any gap. Random jammer alternates between sleeping and jamming. During its jamming phase, it can either behave like a constant jammer or a deceptive jammer. Reactive jammer stays quiet when the channel is idle, but starts transmitting a radio signal as soon as it senses activity on the channel.

Ye et al. [32] analyzed the attacks which deny channel access causing congestion in mobile ad hoc networks. They showed that Mac layer fairness is necessary to reduce the various types of DoS attacks. The factors which decide the efficiency of attack are traffic patterns generated by an attacking node, its location in the network, location of other compromised nodes in the network. The JACK attack that we focus on can be considered as from reactive jammers. The difference is that the JACK attackers do not jam the data packets but just the expected ACK packets.

Goldsmith and Wicker [35] summarized several wireless attacks discovered by other researchers and implemented them with the help of an “aux port”, an unbuffered unsynchronized raw memory access interface for debug purpose. Most of the implemented attacks were shown to be successful, underlining the necessities to detect any of these attacks when such networks are employed in mission critical applications.

Chapter 3

The SCTS and CSD Schemes

3.1 Investigation of Spurious CTS Attack

In the IEEE 802.11 MAC protocol, both virtual carrier-sense and physical carrier-sense functions are used to reduce the probability of collision on the shared wireless channel. A node can only send packets when both of these two functions indicate that the medium is idle. Network Allocation Vector (NAV) serves as the key for the virtual carrier-sense function.

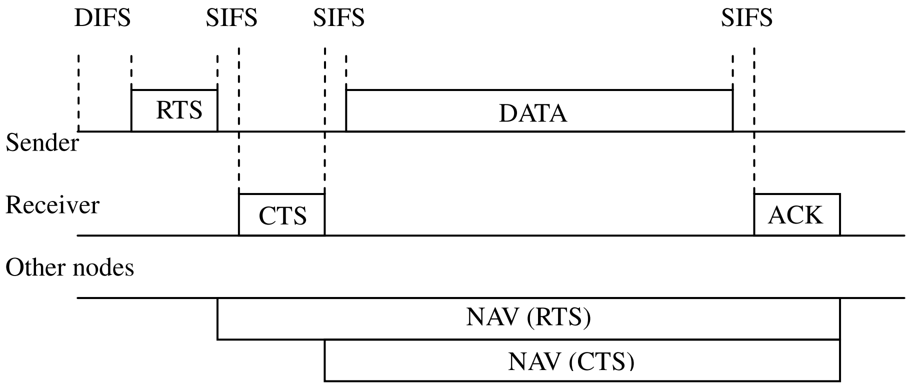

Figure 3.1 The RTS/CTS/DATA/ACK access mechanism

The detail of RTS/CTS/DATA/ACK scheme is depicted in Figure 3.1. The sender sends an RTS packet after waiting for DIFS time. The intended receiver replies with a CTS packet after waiting for SIFS time if the carrier is free. The sender then starts data packet transmission after waiting for SIFS time. The receiver, after receiving the packet, waits for another SIFS time and sends an ACK packet. As soon as the transmission is over, the NAV in each node marks the medium as free and the process can repeat. (DCF inter-frame spacing (DIFS) is used by stations that are operating under DCF mode to transmit packets. Short inter-frame spacing (SIFS) is the shortest of all the IFSs and denotes highest priority to access the medium).

The RTS/CTS packet contains a duration field, which is used to set the NAV of the nodes overhearing the RTS/CTS packet. Every node overhearing an RTS or CTS packet will set its NAV accordingly. The NAV specifies the earliest time at which the node is permitted to attempt transmission. For example, the neighboring nodes of the RTS sender will set their NAVs according to the overheard RTS packet. Similarly, the neighboring nodes of the intended receiver will set their NAVs according to the overheard CTS packet. Note that these two sets of nodes may be different due to their different physical locations. This mechanism protects the transmission between the sender and the intended receiver from any transmission by these neighboring nodes.

In IEEE 802.11 networks when a channel is reserved by a node for transmission, neighboring nodes cannot access the channel though it is idle, until the reserved time expires. Spurious CTS attack exploits the vulnerability (that nodes do not check NAV's validity) denying the channel accessibility to neighboring nodes though the channel is idle. This is because they do not know if the transmission is actually happening during the expected duration. Since a node must defer its transmission when it overhears an RTS/CTS packet, it will be falsely blocked if the corresponding transmission does not take place and the shared medium is left idle. In this case, the shared channel's status is actually idle but its NAV value is still greater than zero.

So far the IEEE 802.11 has the capability to protect wireless networks from spurious RTS attacks, in which a malicious node generate a fake RTS with a high fake NAV value. The protection mechanism is that: an exposed node overhears a RTS sets the NAV value as $(2 \times \mathit{SIFSTime}) + \mathit{CTSTime} + (2 \times \mathit{SlotTime})$. However it should be re-confirmed by the following DATA packet. If there is not any DATA packet the exposed node will reset its NAV later. So under the attack of spurious RTS, the exposed nodes will finally be recovered although waiting for such a long idle period wastes the bandwidth. Since lots of researches [20, 21] based on spurious RTS attacks were explored before, in this thesis we focus on the spurious CTS attacks which have never been touched by now.

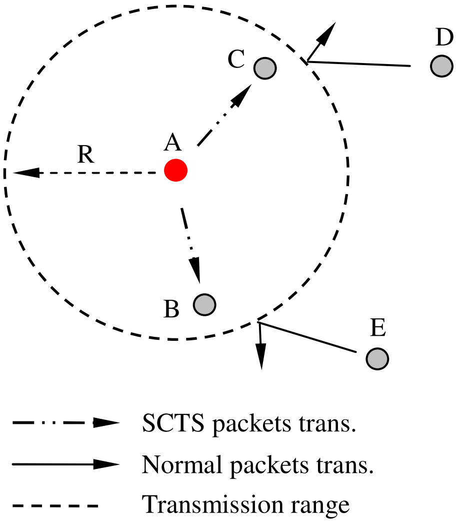

Figure 3.2 The SCTS attack mechanism

In Figure 3.2, Nodes B and C cannot reply to the RTS packets from E and D, because the NAV of SCTS has not yet expired. An attacking node may launch Denial of Service (DoS) [20] attacks by sending out spurious CTS packets periodically. The worst situation is that the attacked nodes are completely blocked when a new spurious RTS/CTS packet is heard prior to the expiration of the NAV. Figure 3.3 illustrates the detail of the SCTS attack in term of NAV duration.

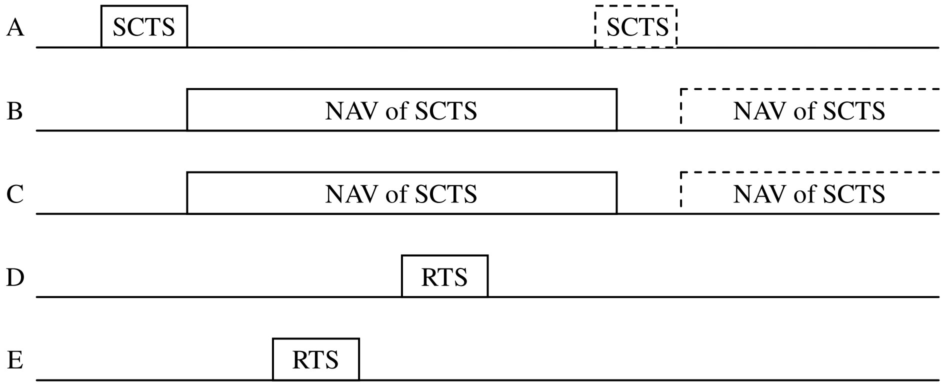

Figure 3.3 Periodical SCTS attack

3.2 Carrier Sensing Based Discarding (CSD) mechanism

Normally, the NAV value implies the busy status period of shared channel. When NAV value bigger than zero, the channel status must be busy and vice versa (NAV value should be zero when channel status is idle). However, under SCTS attacks, the above theorem is broken, because NAV value is bigger than zero while channel is idle. That result in the emergence of the solution: carrier sensing based discarding mechanism.

As described above, when nodes are blocked by SCTS attacks, the shared medium remains unused and wasted. The mechanism of the CSD scheme can be explained as follows: *a node overhearing a CTS packet will assess the status of the channel at the time when the corresponding data packet transmission should start. If the medium within the carrier sensing range is idle, indicating no transmission on the channel, the CTS packet is treated spurious and the corresponding NAV value is discarded. On the other hand, if the channel is sensed busy, the CTS packet is treated as legitimate*. Note that the correctness of such SCTS detection scheme depends on the long carrier sensing range. In the IEEE 802.11 standard, the carrier sensing range is *2.2R*, where *R* is the wireless transmission range [38]. This makes sure that the signal of data transmission from the sender can be sensed by any node that is within a distance of *2.2R* from the sender (cf. Figure 3.4).

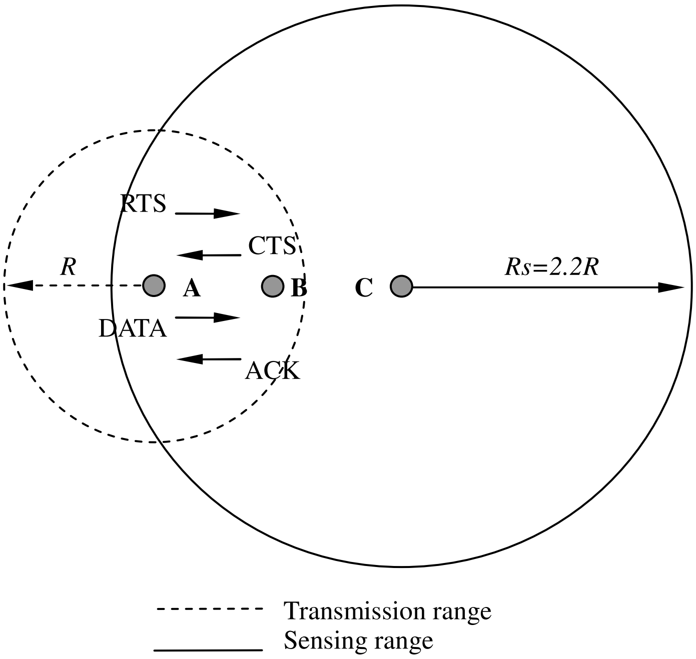

Figure 3.4 Sensing range and transmission range

Under protection of CSD, random channel assessment mitigates the effect of SCTS attacks.

In Figure 3.5, node C, D and E detect the spurious CTS when they assess the idle channel. Meanwhile, node B without the protection of CSD keeps waiting till the fake NAV of SCTS expires.

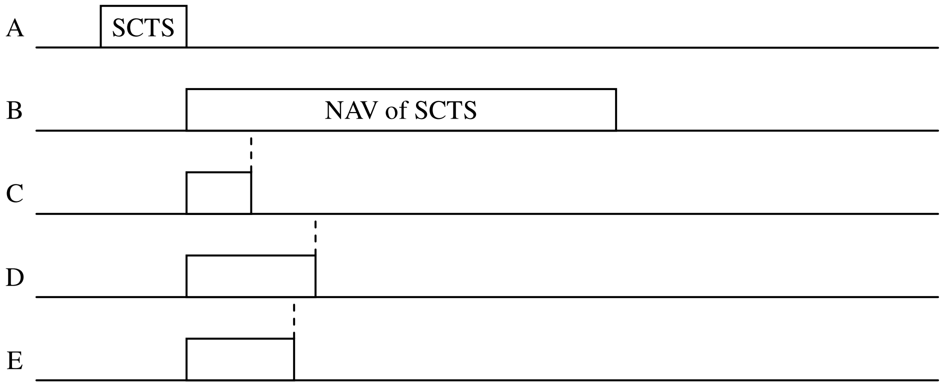

Figure 3.5 Simple CSD mechanism

However, encountering smart SCTS attacks the previous CSD mechanism is very fragile. An intelligent attacker may send bursts of signals to avoid being detected by the CSD scheme. The bursts of signals can be any small control packets, such as RTS, CTS, and ACK, or any meaningless noise. The goal is cheating carrier sense function to generate the illusion that the shared channel is busy. Because in the simple CSD mechanism it just assesses the channel status once, the probability of detection point knocking at the deceptive signals is higher when the intelligent SCTS attacker is smarter.

In Figure 3.6, under intelligent SCTS attack from node A, node E successfully detecting the spurious attack continues RTS/CTS handshake for the normal data transmission. But unfortunately, because node C and D’s detection points are covered by deceptive signals generated by, the simple CSD mechanism fails. They have to waste the precious bandwidth on waiting the expiration of the fake NAV.

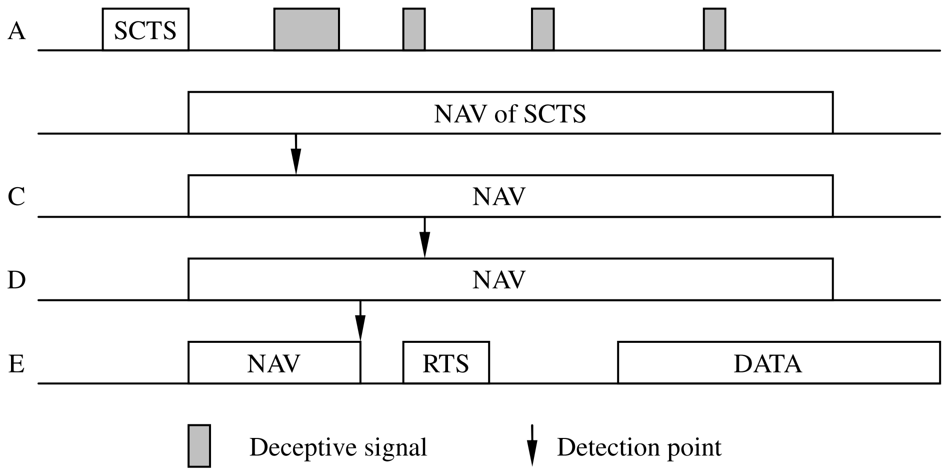

Figure 3.6 Intelligent SCTS

Straightforwardly, in order to protect the network from such intelligent attackers, the number of carrier sensing points of CSD is more than one, and they are chosen randomly among the entire expected data packet transmission time. We design the CSD scheme to sense the carrier on the shared channel *m* times. These *m* detection points work in the following way: if any of these *m* detection points reveals an idle channel, the CTS packet in question is declared as SCTS and the NAV is reset. This process is also known as the OR fusion rule in distributed detection [42]. Other rules such as *k* out of *m* rules may be possible, but we leave those to our future work. An illustration of the CSD scheme is shown in Figure 3.7.

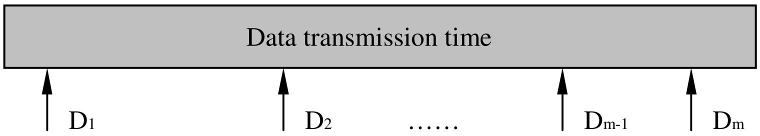

Figure 3.7 Randomly distributed detection points

The operational details of the CSD scheme are presented below:

1) When a node overhears a CTS packet, it computes the DATA transmission time based on the NAV value of the received CTS packet. It is defined as:

$$T_{Data} = T_{NAV} - 2T_{SIFS} - T_{ACK},$$

where T_{Ack} *is the Acknowledgment (ACK) packet transmission time and* T_{SIFS} *is the duration of SIFS.*

2) We generate m detection points, D_i *(i=1,2,...m), which are chosen randomly within the period of* T_{Data} *. We assume that they are in increasing order (cf. Figure 3.7).*

3) At each detection point D_i *(i=1, 2, ...m), the node assesses the channel with a result of* $S(D_i)$*, where* $S(D_i)$ *=1 means channel is busy and* $S(D_i)$ *=0 otherwise. For any I if* $S(D_i)$ *=0, the CSD process is terminated and the CTS packet is declared spurious. The NAV is then reset. Otherwise, increment* i *and repeat.*

4) When all detection points return their decisions as busy channel, the CTS packet is considered normal.

3.3 Analysis of the CSD mechanism

The correctness of the CSD scheme is based on the long carrier sensing range (of *2.2R*). However, besides tackling the intelligent spurious CTS attack, we still need to consider the case in which other nodes within the sensing range of the victim node send packets, affecting the carrier sensing results. This is because a CSD failure occurs when there are transmissions taking place occasionally on all detection points. The Figure 3.8 illustration a missed detection simple: Node B sends a SCTS packet in its neighborhood. If one or more nodes within *2.2R* distance of node A send packets at each of the *m* detection points, CSD fails to detect the SCTS packet.

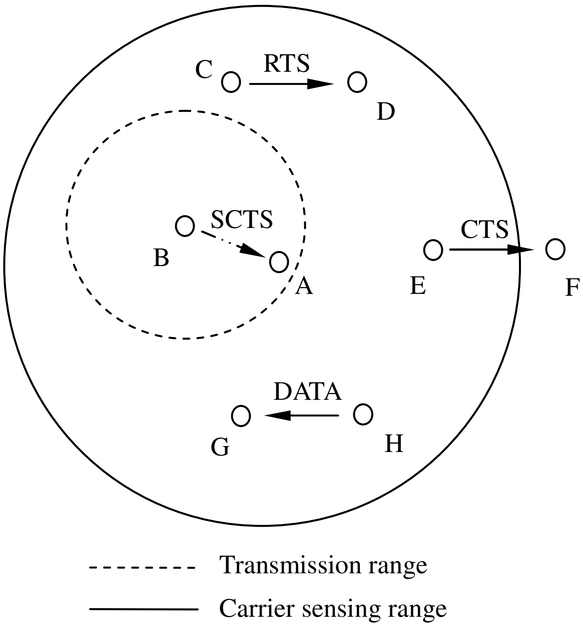

Figure 3.8 Missed detection (1)

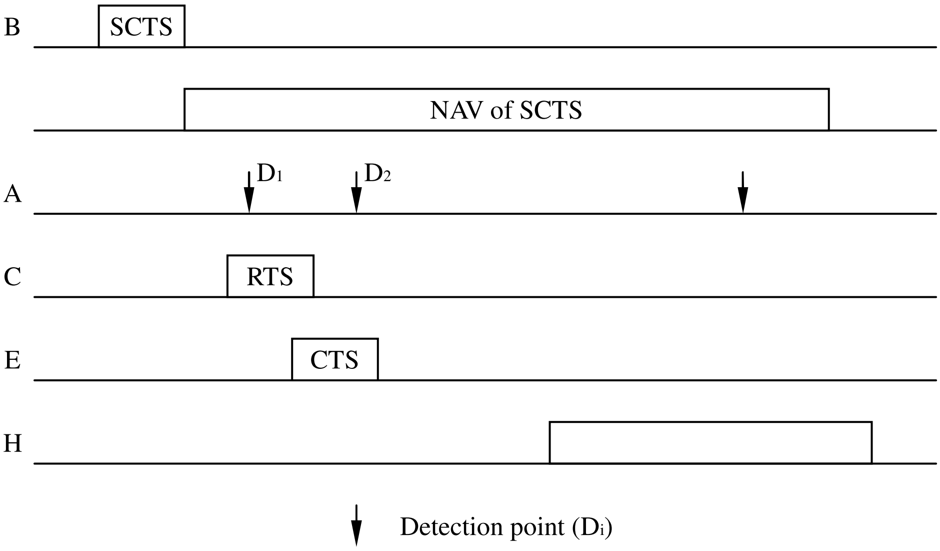

Figure 3.9 Missed detection (2)

In the case described as Figure 3.9, there are three detection points. The CSD fail because in each detection point *S(D*$_i$*)* is true. We compute the probability of missed detection in the CSD scheme, i.e., CSD failure, in the following.

We first introduce our assumptions for analysis: The number of packet transmissions taking place during one unit time within the sensing range of a node is assumed to be Poisson distributed with average packet arrival rate of *G* per unit time. We also assume that these packets have an average duration of τ , and the CSD mechanism has *m* randomly selected detection points and the period of each NAV of SCTS is *T*.

The duration of busy period due to packet transmissions in *T*, T_{Busy} , can be determined by summing the durations of all packets transmitted during *T*. However, there may be packet overlaps as illustrated in Figure 3.10, where packets 2 and 3 overlap and packets (*i-1*) and *i* overlap. Therefore, when the time between the beginning of two consecutive packets is less than τ , they overlap.

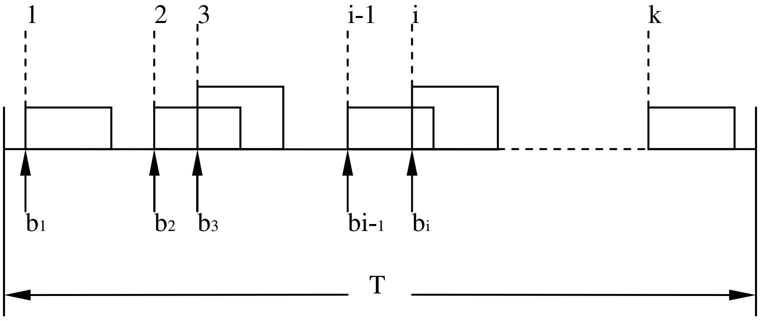

Figure 3.10 Overlapped packets

Denote $t_i = b_i - b_{i-1} < \tau$, so the overlap time is $\tau - t_i$ between packets (*i-1*) and *i*. Based on the Poisson packet arrivals, the expected overlap time between a packet and the next packet can be expressed as:

$$\varepsilon = \int_0^{\tau} (\tau - t) \cdot G \cdot e^{-G \cdot t} dt \qquad (3.1)$$

Note that the calculation of ε is an approximation, which ignores the possibility of more than two packets overlapping.

We argue that this is a good approximation when *G* is relatively small. We will investigate the value of T_{busy} when *G* is large in next section.

When there are *k* packets arriving in *T*, the total busy period is:

$$T_{busy}(k) = k \cdot \tau - (k-1) \cdot \varepsilon \qquad (3.2)$$

Therefore, the average value of the busy period is:

$$T_{busy} = G \cdot T \cdot \tau - (G \cdot T - 1) \cdot \varepsilon \qquad (3.3)$$

where we have assumed an average of $G{\cdot}T$ packets during T units of time.

We assume that an SCTS packet has been sent. The probability of one detection point detecting busy status, which fails to detect SCTS, is:

$$q = \frac{T_{busy}}{T} \qquad (4.4),$$

When there are m detection points, a missed detection (of SCTS packets) occurs if all detection points sense the channel to be busy. The probability of a missed detection, P_{MD}, is then given by:

$$P_{MD} = q^m = [\frac{G \cdot T \cdot \tau - (G \cdot T - 1) \cdot \varepsilon}{T}]^m \qquad (5.5)$$

where ε is given by (3.1).

The analysis of false alarms (or false positives) of SCTS detection in the CSD scheme is trivial. This is because any detection point will sense the busy channel due to the actual data transmission taking place assuming that the sensing mechanism is perfect. Therefore, the probability of false alarm of the CSD scheme is 0.

3.4 Performance evaluation of CSD

3.4.1 Introduction of network simulator (NS2)

NS (version 2) is an object-oriented, discrete event driven network simulator developed at UC Berkely written in C++ and OTcl (Tcl script language with Object-oriented extensions). It implements network protocols such as TCP and UPD, traffic source behavior such as FTP, Telnet, Web, CBR and VBR, router queue management mechanism such as Drop Tail, RED and CBQ, routing algorithms such as Dijkstra, and more. NS also implements multicasting and some of the MAC layer protocols for LAN simulations [12].

NAM is a Tcl/TK based animation tool for viewing network simulation traces and real world packet trace data. The first step to use NAM is to produce the trace file. The trace file should contain topology information, e.g., nodes, links, as well as packet traces. Usually, the trace file is generated by NS [39]. During an ns simulation, user can produce topology configurations, layout information, and packet traces using tracing events in NS.

When the trace file is generated, it is ready to be animated by NAM. Upon startup, NAM will read the trace file, create topology, pop up a window, do layout if necessary, then pause at the time of the first packet in the trace file. Through its user interface, NAM provides control over many aspects of animation.

To model a Network simulating using NS2 is necessary to write a Tcl script describing the topology (nodes, agents, applications, etc.) [13].

3.4.2 Accuracy of T_{busy} and P_{MD}

We present our simulation results for T_{busy} and compare them with our analytical results in (3.4). We used $\tau = 1$ second and T = 100 seconds. As can be seen from Figure 3.11, these two sets of results match well with each other. As G increases further, T_{busy} approaches T.

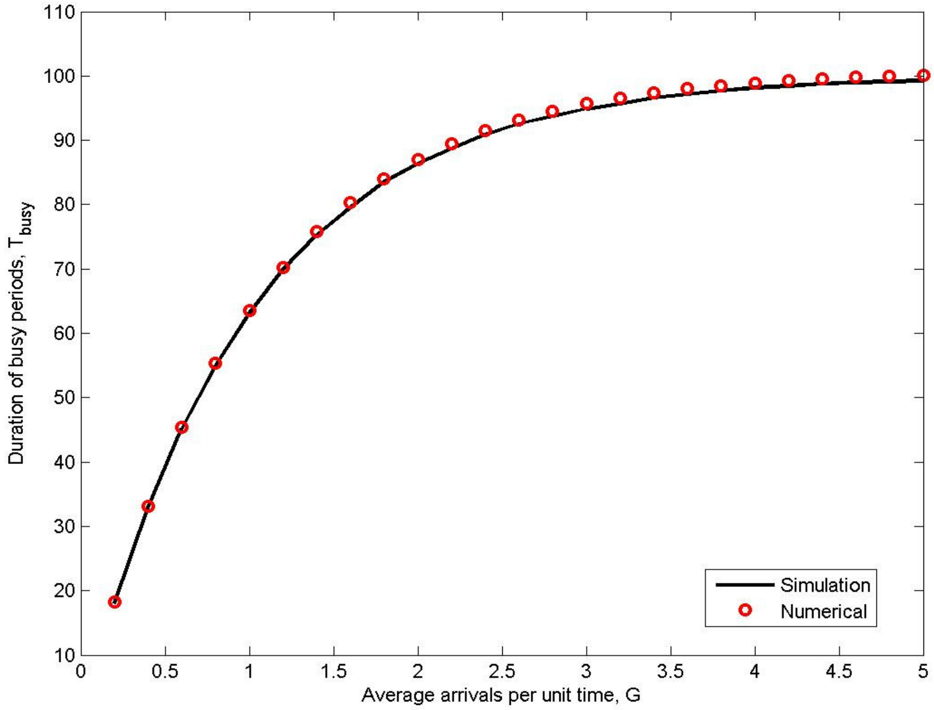

Figure 3.11 Comparison between simulation results and numerical results of busy time

In order to demonstrate the effects of the number of detection points, m, on missed detection probability, P_{MD}, we present Figure 3.12. The data packet transmission time is 603 msec and the period of each NAV of SCTS, T, is 4.55 sec. Based on Figure 3.12, P_{MD} increases with G. When m increases, P_{MD} reduces because of higher chances of detecting SCTS packets. Our simulation results match well with numerical results except in very large G regions, where the assumption of only two packet collisions taking place becomes invalid. That is why simulation values are always larger than numerical values and difference increases with G.

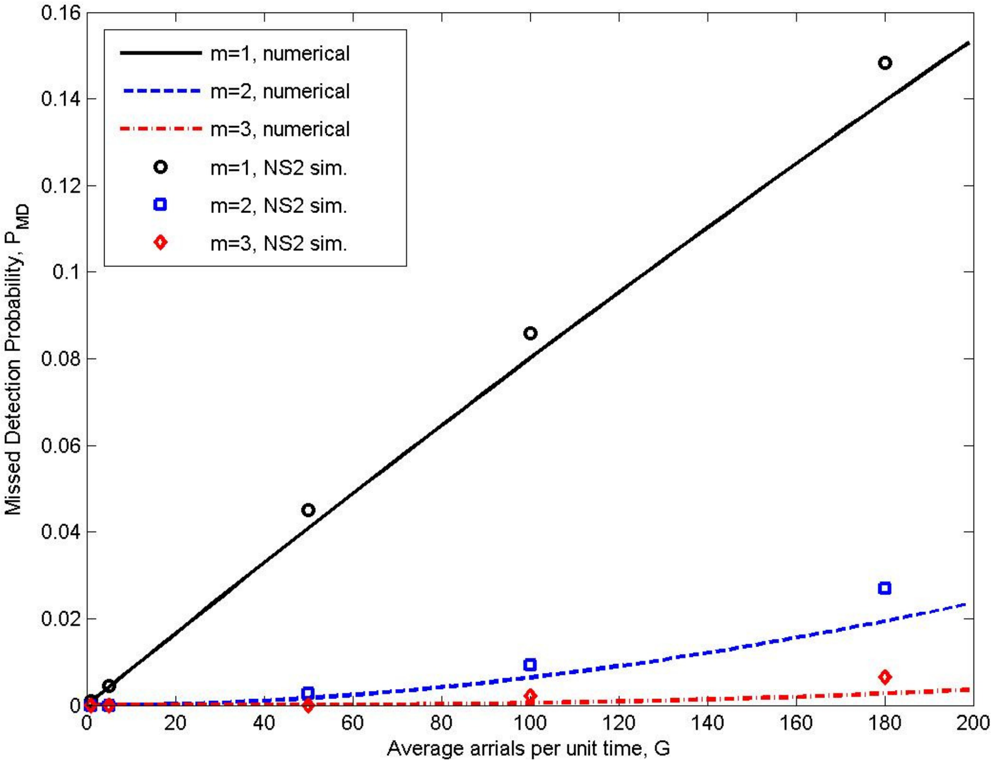

Figure 3.12 simulation and numerical results of missed detection probability

3.4.3 Performance of CSD

We used ns2 [44] to simulate several IEEE-802.11b-based MANETs and to investigate the effect of SCTS attacks and that of our CSD scheme. In our simulation, we assumed that all SCTS packets tried to reserve the use of the channel for the maximum possible time, which we identified as 45.5 msec. Another important parameter for the malicious SCTS packet senders is the average interval between two consecutive SCTS packets. We term such interval ASI (average duration of SCTS Intervals). Unless specified otherwise, ASI = 65.3 msec.

Some critical parameters are list in the following table 3.1.

Table 3.1

Tcl Interface Configurations of three-node model

Description	Parameter	Value
channel type	chan	Channel/WirelessChannel
radio-propagation model	prop	Propagation/TwoRayGround
network interface type	netif	Phy/WirelessPhy
MAC type	mac	Mac/802_11
interface queue type	ifq	CMUPriQueue
link layer type	ll	LL
antenna model	ant	Antenna/OmniAntenna
max packet in ifq	ifqlen	100
size of packet	simCBRPktSize	1000
data rate	dataRate_	11Mb
basic data rate	basicRate_	5Mb

To find the relationship between ASI and the throughput of the network, we study a simple fixed 3-node wireless network first (cf. Figure 3.13) with fixed CBR. Reducing ASI increases the frequency of sending malicious CTS packets that results in throughput decrease.

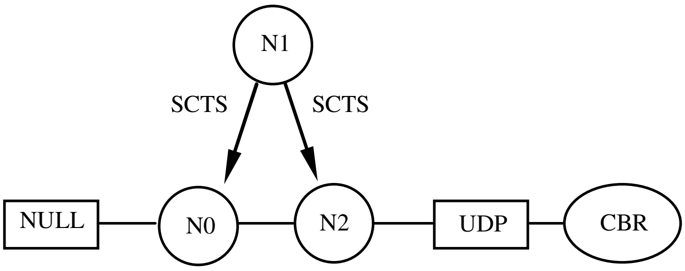

Figure 3.13 The ns2 model of fixed three-node wireless network

In Figure 3.14 we show the relative throughput compared with that of a network without SCTS attackers. Naturally, the maximum possible value of relative throughput is 1.

We investigate four cases, where the packet generation rates of CBR are 0.4 Mbps, 0.6 Mbps, 0.8 Mbps and 1.0 Mbps, respectively. The SCTS packets cause more negative effect with higher traffic load. As ASI increases, throughput improves.

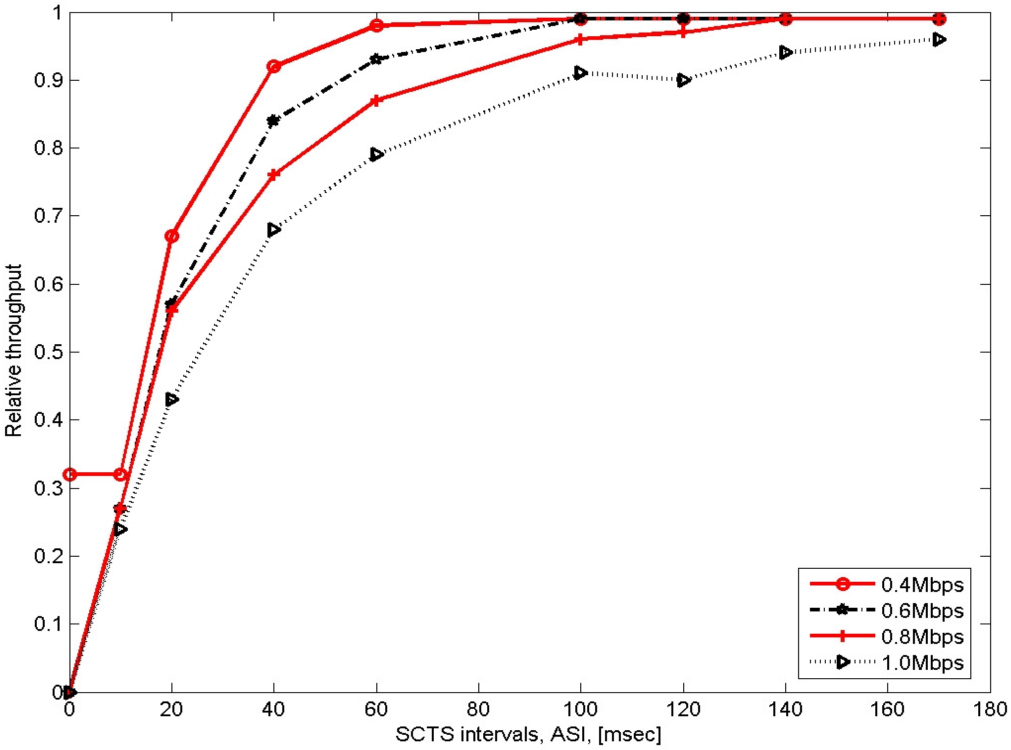

Figure 3.14 Effect of ASI on the throughput in a 3-node network. There are SCTS attackers but CSD is inactive

We compared the throughput (S) of a normal network (without SCTS), one with SCTS, and one with SCTS and CSD under different traffic loads in the fixed three-node wireless network in Figure 3.15.

Before the throughput tends to saturation at traffic load 3.5 Mbps, SCTS attackers cause more damage with increased traffic-load settings (L). That is because higher traffic load means higher probability for attackers competing for more unfair access of the shared channel. And with the CSD scheme, the network throughput can be restored effectively.

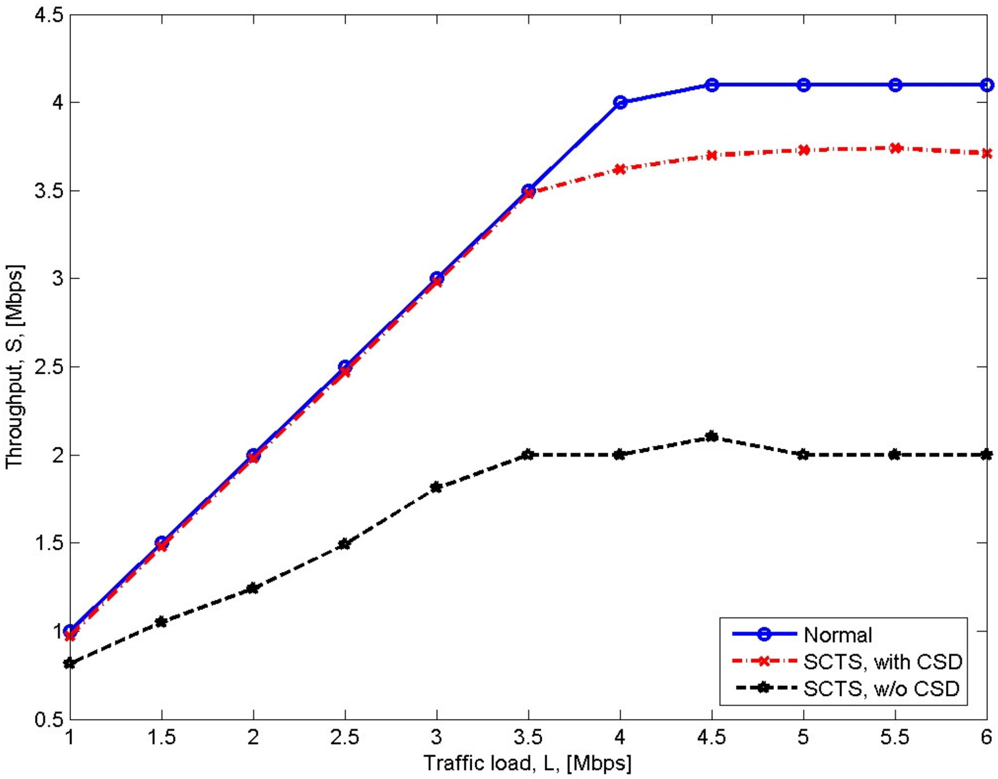

Figure 3.15 Throughput comparisons between the SCTS and the CSD (3-node model)

We also set up a 100-node wireless network over a 1000 × 1000 region. These nodes are distributed randomly (cf. Figure 3.16).

There are 10 random sender/receiver pairs. We investigated the network throughput obtained under SCTS attacks and compared the effect of SCTS and CSD mechanism by increasing the number of spurious nodes. The NS2 simulation is configured as table 3.2.

Table 3.2

Tcl Interface Configurations for 100-node Model

Description	Parameter	Value
channel type	simCBRPktSize	512 bytes
data rate	dataRate_	2.0 Mbps
base data rate	basicRate_	2.0 Mbps
max packet in ifq	ifqen	50
number of node	nn	100

The results are shown in Figure 3.17. Obviously, more SCTS attackers cause more severe degradation of network throughput. With CSD, the network throughput is restored to 85% level of a similar network without SCTS attackers.

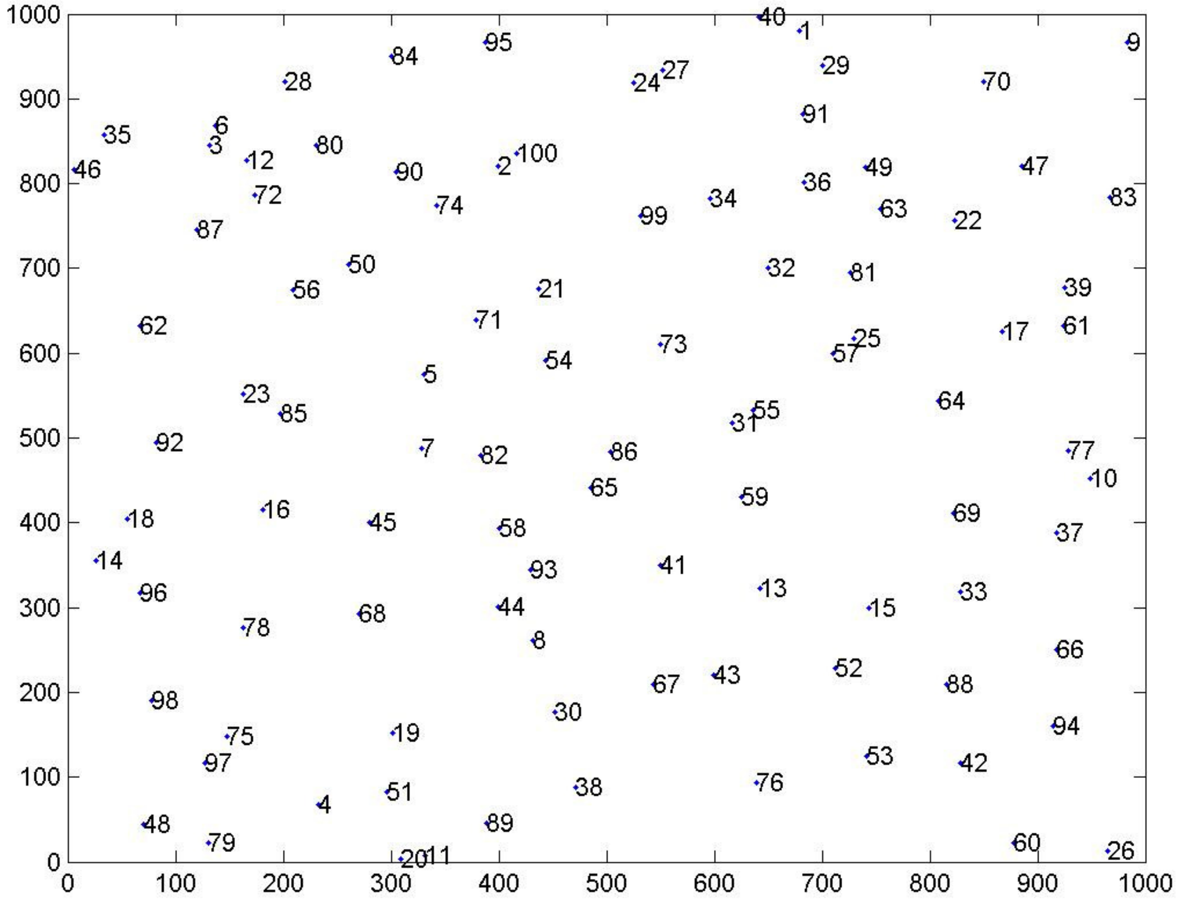

Figure 3.16 100-node wireless network model

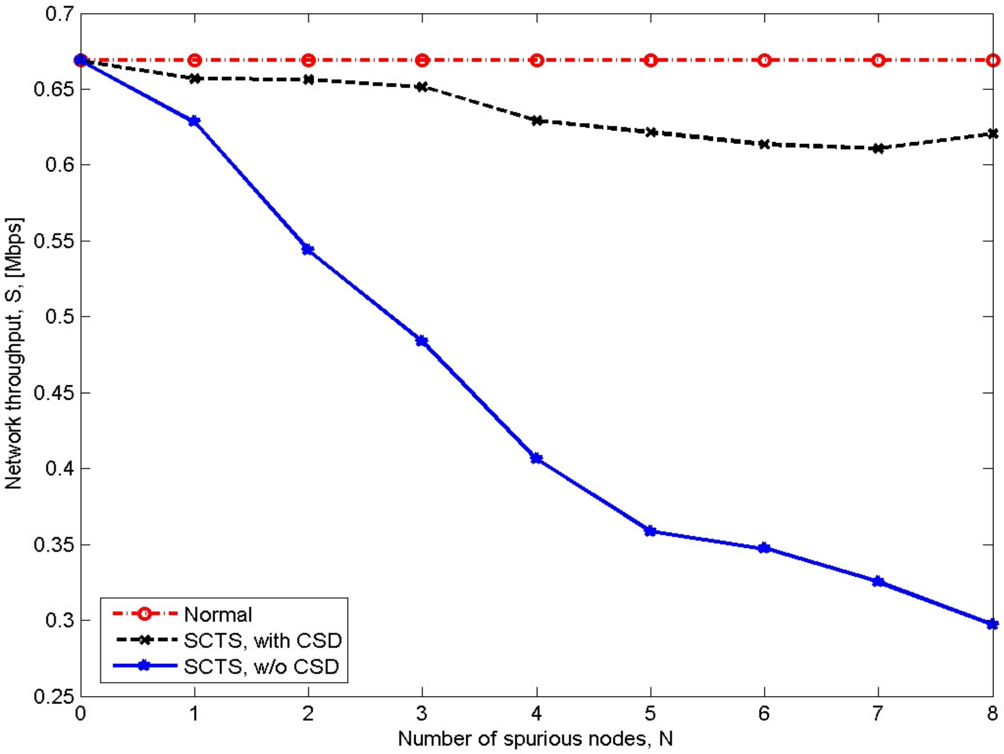

Figure 3.17 Throughput comparisons of the SCTS and the CSD (100-node model)

The effect of the SCTS and CSD approach with different number of detection points is investigated in Figure 3.18. As *m* increases, the performance of the CSD scheme gets better and approaches the performance of a similar network without SCTS attackers. The cost of a larger *m* in the CSD scheme is the increased memory usage and CPU resources at the detecting nodes.

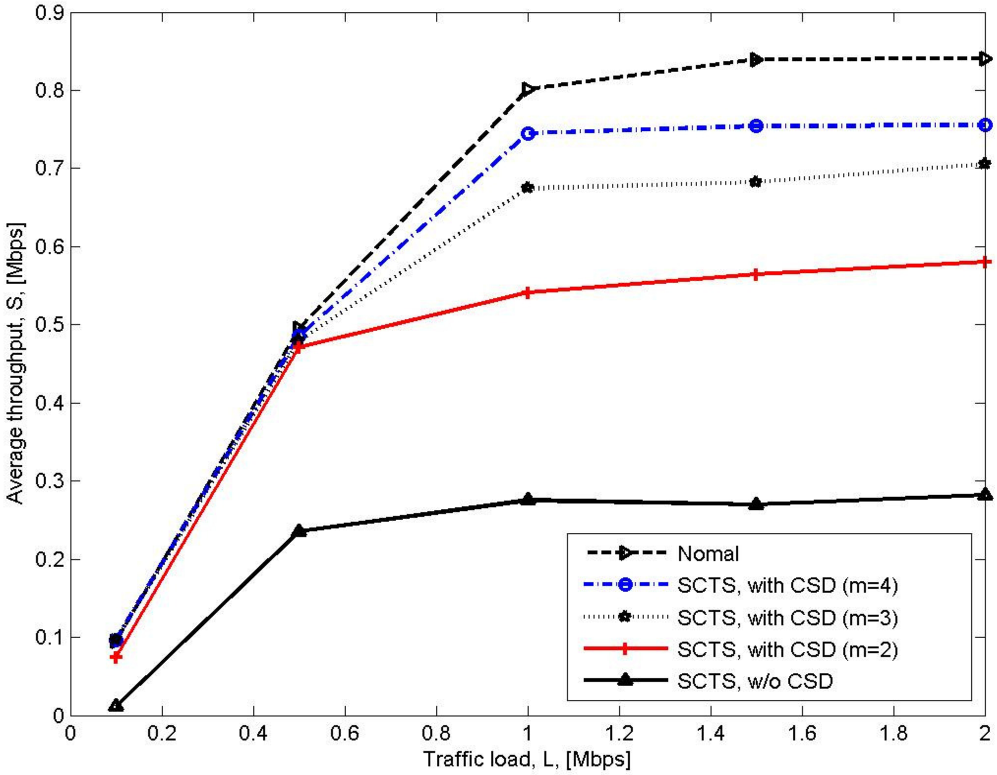

Figure 3.18 Effect of different *m* on SCTS and CSD

3.5 Conclusion of SCTS and CSD mechanism

The RTS/CTS mechanism combined with the NAV scheme is currently used to avoid packet collisions caused by hidden nodes in many ad-hoc network MAC scheme such as IEEE 802.11 DCF. Unfortunately, this leads to the vulnerability of the virtual carrier-sense function: misbehaving or malicious users may send spurious packets especially spurious CTS packets to block other users from accessing the channel. Due to the inherently vulnerable design of the IEEE 802.11 DCF scheme, the attackers are able to block such channels with only very limited number of packet transmissions (as compared to physical channel jamming).

In this chapter, we have proposed the Carrier Sensing based Discarding (CSD) scheme to mitigate such adverse effects of the spurious CTS packets. Instead of asking each node overhearing a CTS packet to update its NAV value, the CSD scheme requires a node to check the validity of the on-going data communication during the entire period of data packet transmission time. Such carrier sensing is possible thanks to the larger carrier sensing range in

IEEE 802.11 DCF (*2.2R*). We have presented the technical details of the CSD scheme and our analysis. Simulation results show that the CSD scheme recovers most of the channel throughput in networks under spurious CTS attacks as compared to regular networks.

Chapter 4

JACK and ENAV Schemes

4.1 Investigation of The Jamming ACK Attack

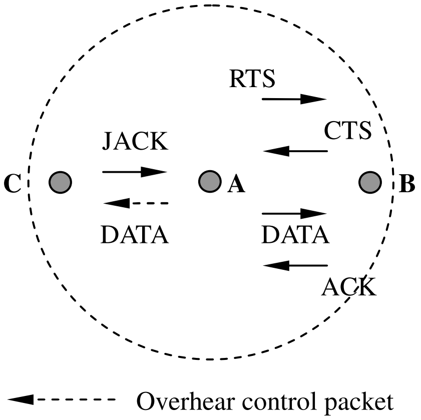

Figure 4.1 JACK attack mechanism (1)

Based on the operational rules of the MAC schemes, such as IEEE 802.11 DCF, in MANETs, all data packets need to be acknowledged before they are cleared from the queue. An attacker simply sends wireless signal to jam ACK messages when it overhears a DATA packet in the network. Such jamming signal ruins the reception of ACK message at the sender of the data packet as long as the attacker locates within the range of the transmitter. This is illustrated in Figure 4.1.

We illustrate the details of JACK in the Figure 4.2. When the ACK packet is jammed by attackers, data retransmissions will be scheduled. Such retransmissions will fail in a similar fashion. Data packets will be simply dropped once the sender reaches the retransmission limit.

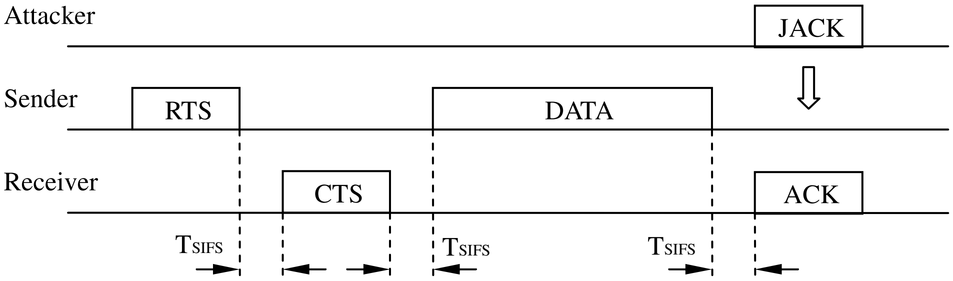

Figure 4.2 JACK attack mechanism (2)

An interesting observation to the victims of such attack is that they consume more energy in vain in order to make sure that all data packets are transmitted successfully, i.e., ACK packets are expected to arrive after successful data transmission. Hence, the attackers effectively cost the victims extra energy by jamming a short control packet, the ACK packet.

For example, in the IEEE 802.11 DCF scheme, the default retransmission limit is three times. That means the RTS/CTS/DATA/ACK process is repeated three times on retransmitting the same data packet when each acknowledgment is ruined by a short malicious signal (in this case which is an ACK packet). Three times extra energy is wasted compared to that in a normal situation.

In addition, such retransmissions block the channel from sending other useful data, resulting in reduction of maximum achievable throughput. Note that the data receiver has already received the DATA packet even if the ACK packet suffers from collisions. We term the potential attack Jamming ACK (JACK) attack. Therefore, the adverse effect of the JACK attacks can be summarized as follows:

- Higher energy cost of the data sender;
- Lower maximum achievable throughput.

The operational details of a JACK attacker are the following: It tries to overhear on the shared channel and wait for any DATA packet from the sender. Once a DATA packet is overheard, it waits for a period of Short InterFrame Spacing (SIFS) and sends out the JACK packet. In fact, any packet sent by the JACK attacker ruins the reception of the legitimate ACK packet. (In general, the JACK attacker does not need to wait for a period of SIFS time.

Because in the IEEE 802.11, the transmission time of ACK packet is 200ms and T_{SIFS} is 10ms. When a spurious ACK packet is sent as soon as overhearing an entire data packet, it can guarantee the ACK collision. But to simplify the JACK attack we ignore the small period of T_{SIFS}).

There are some reasons why JACK chooses that mechanism:

- Full ACK packet: the size of ACK is relatively small, and it costs small amount of energy for the attacker. In fact, the ACK packet has the smallest size among the RTS/CTS/DATA/ACK packets (CTS packet is also the same size).
- Passive jamming: if we adopt a high frequency to repeat transmission of ACK packets, partial packets, or jamming signals from the JACK attacker may cause suspicion from other nodes. For instance, two consecutive ACK packets from the same receiver usually have an interval of at least one DATA packet transmission time. On the other hand, with low frequency, it performs a less efficient attack. The most optimal opportunity is triggered by the attempt of data transmission of a sender.
- Overhear DATA packet: both control packet RTS and CTS have Duration/ID field to imply an excepted reserved period for the data transmission. But JACK attack cannot be triggered by either of them. The reason is: first, overhearing a RTS packet is not reliable in case that a sender's RTS request cannot be replied by the intended receiver because the NAV is not expired (when it had already overheard a RTS/CTS before and had processed as an exposed/hidden terminal) or the RTS cannot reach the intended receiver because data collision maybe happen. Second, a JACK attacker as an exposed terminal of a sender cannot overhear the CTS generated by the receiver (because it is out of the transmission range of the receiver). Figure 4.3 illustrates the case that node B's RTS cannot reach node C or node C cannot reply a CTS because it should keep silent treated as an exposed/hidden terminal. Because node A is out of the transmission range of node C, it cannot overhear the CTS packet from node C.

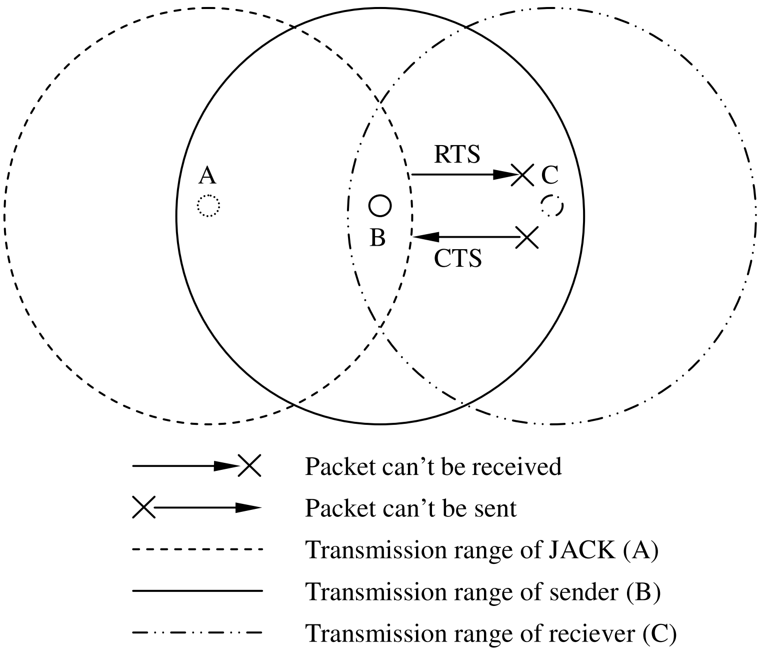
A
B
C
RTS
CTS
Packet can’t be received
Packet can’t be sent
Transmission range of JACK (A)
Transmission range of sender (B)
Transmission range of reciever (C)

Figure 4.3 RTS/CTS is unreliable for JACK

Note that jamming other packets may not lead to such a high energy drainage from the victim nodes. For instance, jamming the RTS/CTS packets only leads to retransmission of such control packets, which is less effective compared to the retransmission of data packets.

4.2 The ENAV Scheme

In this section, we introduce a scheme to mitigate the adverse effect of JACK attackers. The main idea of our scheme is an extension of the ACK transmission window and random transmission time over this period. This technique is termed Extended NAV (ENAV), illustrated in Figure 4.4. In this figure, the ACK transmission window is extended from T_{ACK} to $R \cdot T_{ACK}$. By extending the window of sending/receiving the ACK packet, the data sender has a better chance of receiving the ACK packet from the data receiver.

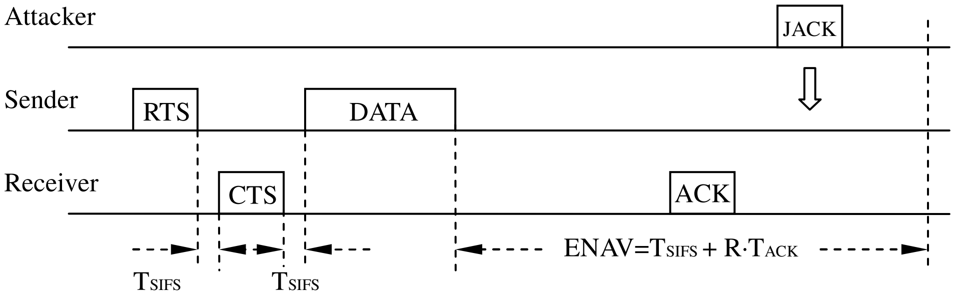

Figure 4.4 Illustration of the ENAV scheme

Obviously, when R = 1, a MAC scheme implementing the ENAV scheme degenerates to the original MAC scheme.

While the NAV values carried on the RTS and the CTS packets change from one transmission to another due to the variable DATA packet length, the NAV value carried on DATA packets is usually fixed at $T_{SIFS} + T_{ACK}$. Because a JACK attacker can overhear a complete DATA packet prior to the expiration of the NAV of DATA, it may notice the extension of the NAV value and hence try to send its JACK packet to collide with the real ACK packet in the extended period. However, to conceal himself sedulously from being detected, the JACK attacker still send single spurious ACK packet. Since the legitimate ACK packet is sent randomly within the ACK transmission window the JACK attacker cannot guess when the real ACK packet will be sent from the data receiver. The best option that it has is to send at a randomly-chosen time between [0, (R – 1) · T_{ACK}], the period between the rear edge of DATA packet and expiration of the NAV.

With ENAV, the receiver will delay for a random period within

$$[0,(R-1)\cdot T_{ACK}] \tag{4.1}$$

after the complete reception of DATA packet and a $T_{SIFS.}$

If the sender does not receive the ACK packet, it will keep on retransmitting until success or it reaches the maximum retransmission limit. Let N_{t_max} ($N_{t\_max} \leq 1$) denote the maximum transmission limit of one DATA packet, which means the sender will retransmit it for at most N_{t_max} – 1 times.

4.3 ENAV Performance Evaluation

In this section, we analyze the effect of *R* in the ENAV scheme on throughput. These results are then compared to simulation results from NS2.

4.3.1 Anaylsis of ENAV

Denote the probability that an ACK packet collides with a JACK packet *P(R)* for a given *R* in (4.1). Clearly, *P(R)* = 1 if $R \leq 2$. So we focus on the situation where $R > 2$.

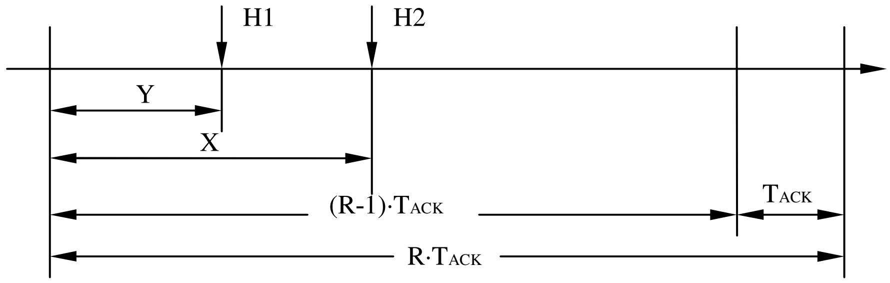

Figure 4.5 Derivation of collision probability in the ENAV scheme, *P*(*R*)

Assuming that both the ACK and the JACK packets are to be transmitted randomly within this $R \cdot T_{ACK}$ period of time, the beginning of the ACK packet, H1, and the beginning of the JACK packet, H2, will be randomly chosen from the period between 0 and $(R-1) \cdot T_{ACK}$, as shown in Fig 4.5. The Probability Density Function (PDF) of H1 and H2 is:

$$f(x) = f(y) = \frac{1}{(R-1)T_{ACK}} \tag{4.2}$$

Collisions of the ACK and the JACK packets occur if

$$|H1 - H2| = |x - y| < T_{ACK} \tag{4.3}$$

Therefore, the probability that these two packets collide with each other can be calculated as (4.4). We assume $R > 3$ (the $2 \leq R \leq 3$ case is similar but omitted due to space limit).

$$\begin{aligned} P(R) = &\int_0^{T_{ACK}} f(x) \int_0^{x+T_{ACK}} f(y)dydx \\ &+ \int_{T_{ACK}}^{(R-2)T_{ACK}} f(x) \int_{x-T_{ACK}}^{x+T_{ACK}} f(y)dydx \end{aligned}$$

$$+\int_{(R-2)T_{ACK}}^{(R-1)T_{ACK}} f(x)\int_{x-T_{ACK}}^{(R-1)T_{ACK}} f(y)dydx$$

$$=\frac{2R-3}{(R-1)^2}\text{ (when } R>3) \qquad (4.4)$$

We explain the calculation as follows: the overall possible $(R-1)T_{ACK}$ transmission window is divided into three parts (when $R > 3$): $[0, T_{ACK}]$, $[T_{ACK}, (R-2)T_{ACK}]$, and $[(R-2)T_{ACK}, (R-1)T_{ACK}]$. The three terms in (4.4) calculate the chance that x falls in the three parts, respectively, and (4.3) is satisfied.

When n (re)transmissions are sent for one data packet, the overall transmission time of this data packet can be calculated as $T(R, n)$:

$$T(R,n)=T(R,n-1)+B(n)+T_1+R\cdot T_{ACK} \qquad (4.5)$$

where $n \geq 2$, $T_1 = T_{DIFS} + T_{RTS} + T_{CTS} + T_{DATA} + 3T_{SIFS}$, $B(n)$ is the average back-off time in the n-th (re)transmission, and $T(R, 1) = T_1 + B(1) + R \cdot T_{ACK}$.

Considering the probability of collision, for each DATA packet, the average overall transmission time is $T(R)$:

$$T(R)=\sum_{n=1}^{3}T(R,n)\cdot[P(R)]^{n-1}\cdot[1-P(R)]+T(R,4)\cdot[P(R)]^3 \qquad (4.6)$$

We need to derive $B(n)$ in (4.5). $B(n)$ represents the average back-off time of each (re)transmission. Since the back-off timers are chosen randomly from the Contention Window (CW), $B(n) = \mathrm{CW}(n)/2$.

We numerically calculated the throughput of the simple three-node network (see Figure 4.1) with ENAV employed and compared them with NS2 simulations. The parameters of our calculation are listed in Table 4.1.

TABLE 4.1

NS2 Simulation Parameters

NIC: dataRate	11 Mbps	$B(1)$	320µs
NIC: basicRate	4 Mbps	$B(2)$	640µs
CBR: rate	4 Mbps	$B(3)$	1280µs
CBR: packetSize	1 Kbytes	$B(4)$	2560µs

TABLE 4.1 Continue

N_{t_max}	4	T_{RTS}	232μs
T_{slot}	20μs	T_{DATA}	954μs
T_{SIFS}	10μs	T_{CTS}	220μs
T_{DIFS}	50μs	T_{ACK}	220μs

The throughput of a network with ENAV employed can be estimated based on packet length and $T(R)$. One interesting observation of such a network is that data packet is successful in all (re)transmissions. It is because of the JACK attacks and the ACK packet collisions that prompt the sender to retransmit. For example, when the data packet length is L bytes, the throughput can be expressed as:

$$S(R) \approx \frac{8L}{T(R)} \tag{4.7}$$

An optimization of the throughput based on R is possible. Our derivations show that a maximum throughput may be achieved with R = 7.5. Such numerical results will be compared to simulation results in the next section.

4.3.2 NS2 Simulation of ENAV

In order to show the different effects of JACK attack and ENAV scheme. We carried out simulations for the following 4 scenarios.

- (*normal*): network without JACK attack and ENAV scheme.
- (*JACK only*): network with JACK attack but without ENAV scheme.
- (*JACK+ENAV*): network with both JACK attack and ENAV scheme.
- (*ENAV only*): network with ENAV scheme but without JACK attack.

All the remaining network parameters are shown in Table 4.1. Unless specified otherwise, the Constant Bit Rate (CBR) packet size is always 2000 bytes and R in ENAV scheme is 7.

We show the throughput performance of the simple three-node network in Figure 4.6. In the three-node network, a pair of nodes serves as the sender and receiver. The third node is only neighboring to the sender and it serves as the JACK attacker (see Figure 4.1). The throughput of the network with JACK+ENAV is shown as a function of different *R* in the ENAV scheme.

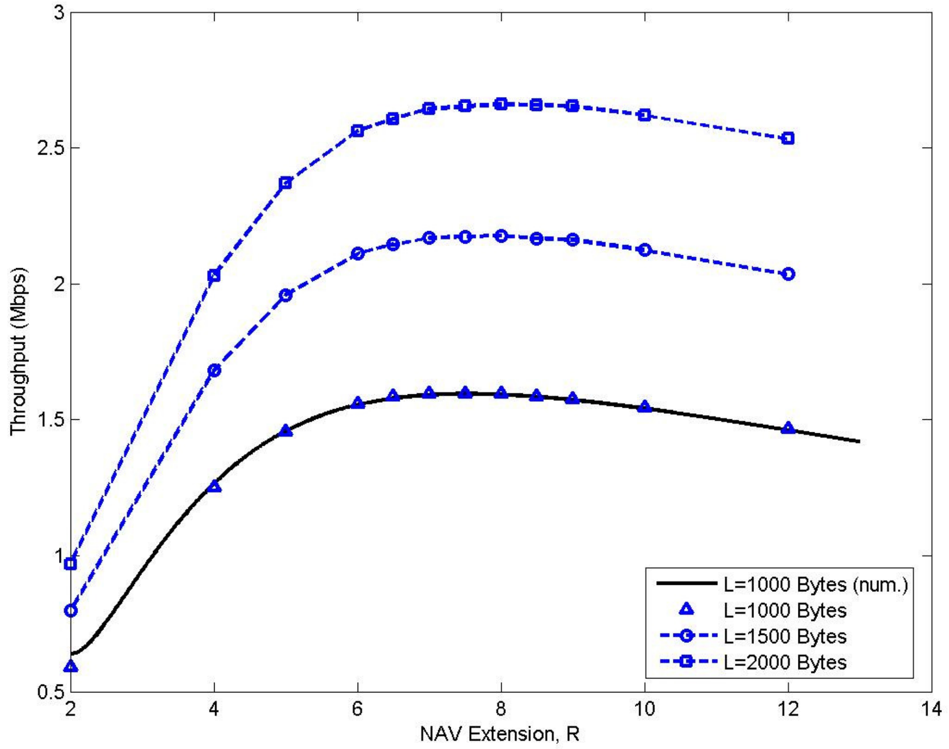

Figure 4.6 JACK and ENAV implementations

A solid line in Figure 4.6 represents our numerical results based on Table 4.1 and equation 4.7. The numerical results match well with NS2 simulation results.

We also present the simulation results of data packet length of 1500 bytes and 2000 bytes. As data packet length increases, the network throughput improves. Based on the simulation results, we can observe that the optimal value of *R* is about 7.5 in the network that we studied. Such an optimal *R* that maximizes the maximum achievable throughput remains the same for different data packet lengths as well.

The simulation results of different traffic load are shown in Figure 4.7. The maximum throughput of a normal network is 6 Mbps. When the network is under JACK attacks, the maximum throughput reduces to 1 Mbps (15% of the throughput of the normal network). With the help of the ENAV scheme, the maximum achievable throughput is recovered to the level of 2.5 Mbps.

For comparison purposes, we also show the throughput of a normal case when the ENAV scheme (with R = 7) is implemented. The throughput of such a network is about 4 Mbps. The lowered throughput is due to the additional channel usage during the extended NAV period in ACK packet transmission/reception. Note that this throughput can be improved with a lowered R, which is possible when the sender/receiver notice no ACK packet loss.

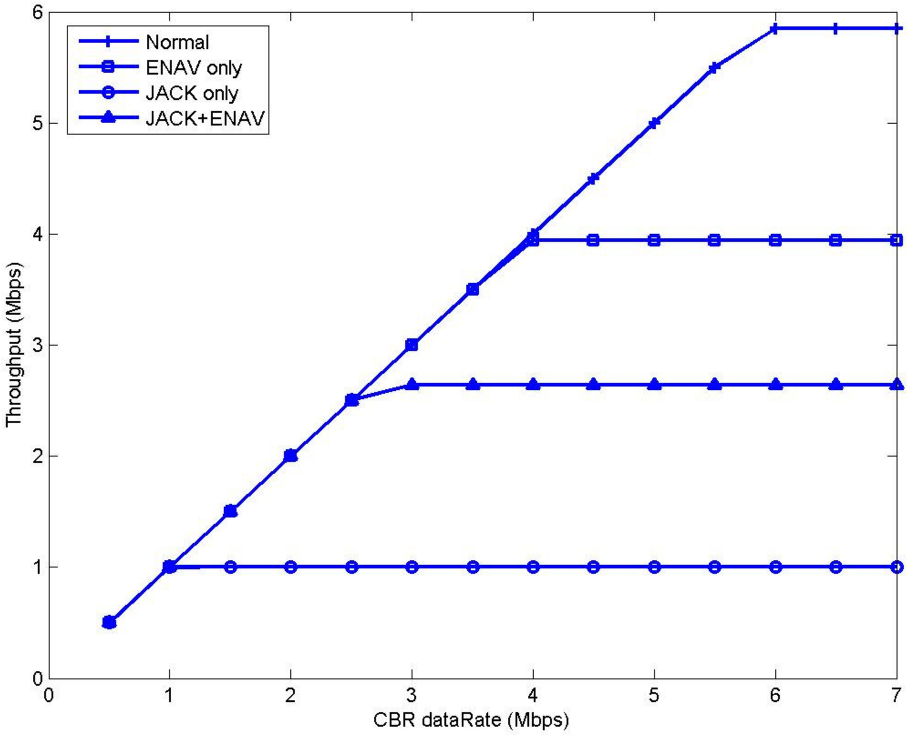

Figure 4.7: Throughput comparison of the normal, the JACK only, the ENAV only, and the JACK+ENAV network

We studied the energy consumption of the nodes in the above four scenarios. The wireless interface cards of sender, receiver, and attacker are assumed to have the specifications as shown in Table 4.2 [30].

TABLE 4.2

Lucent IEEE 802.11 WaveLan PC Card

Transmission Speed	11Mbps
Power Supply	4.74V
Sleep Mode Current	10mA
Sleep Mode Power	47.4mW

TABLE 4.2 Continue

Idle Mode Current	156mA
Idle Mode Power	739.44mW
Receive Mode Current	190mA
Receive Mode Power	900.6mW
Transmit Mode Current	284mA
Transmit Mode Power	1346.16mW

Table 4.3 shows the average energy consumption for each CBR packet in the normal, ENAV only, JACK only, and JACK+ENAV networks. The energy consumption for each packet is the lowest in the normal network since there is no attack or extended NAV. When a network suffers from JACK attacks, the sender and the receiver increase the energy consumption to more than 5 times. In the JACK+ENAV network, the energy consumption of the attacked nodes is reduced to 40% of that of the JACK only network. In the ENAV only network, the energy consumption of the sender and receiver increases slightly from the normal network.

TABLE 4.3

Energy consumption of each data packet transmission. The unit is in MJ

	normal	ENAV only	JACK only	JACK+ENAV
Sender	3.28	4.27	16.86	4.34
Receiver	2.63	3.62	14.23	5.39
Attacker	N/A	N/A	13.67	5.25

4.4 Conclusion of ENAV

With the wide adoption of wireless networks, they are becoming targets of many attacks. At the MAC layer, wireless networks are more vulnerable than wired networks. We have investigated the Jamming ACK (JACK) attack to MAC schemes that require the data receiver to return ACK packets to acknowledge the success of data reception. Such JACK attacks may be launched by adversaries to lower the achievable network throughput and to increase the energy consumption by the victim nodes. Due to its special characteristics, such attackers are difficult to detect or identify. Our study has shown that a JACK attacker can easily raise the energy consumption of a victim sender by 5 times and reduce the achievable throughput of the network to 15%.

We have proposed a solution, termed Extended Network Allocator Vector (ENAV), to mitigate the impact of JACK attacks. With the help of the extended NAV period, the ENAV scheme provides a flexible period for the data receiver to send the ACK packet, significantly reducing the chance of being collided by the JACK attacker. Our analysis and simulations show that the ENAV scheme recovers a significant portion of the network throughput and reduces the energy consumption by the victim nodes to 40%.

Chapter 5

Conclusions

Mobile Ad Hoc Networks (MANETs) have been studied extensively over the past few years, due to their potentially widespread application in military and civilian communications [2]. Such a network is highly dependent on the cooperation of all its members to perform networking functions. This makes it highly vulnerable to selfish and malicious nodes.

In this thesis we have investigated on the security issues on MAC layer to investigate some selfish and malicious behavior: Spurious CTS (SCTS) attack and Jamming ACK (JACK) attack. A SCTS attacker taking advantage of RTS/CTS/DATA/ACK mechanism sends fake CTS packets to instigate its neighboring nodes modify their NAV value for the intention of blocking the normal transmissions. Through jamming the legitimate ACK packet, a JACK attacker causes the normal data packet be retransmitted more times. A small amount of energy consumed by the JACK attacker will drain more energy of the victim on retransmissions.

The solutions of SCTS and JACK have been proposed respectively: Carrier Sensing based Discarding (CSD) and Extended NAV (ENAV). In CSD mechanism, the receiver examines the receiving CTS by randomly deploying some detection points during the subsequent data transmission period. If the channel status keep busy at each detection point, the CTS is valid. Or the CTS is illegitimate, and the NAV register is reset. In ENAV mechanism, the receiver extends the ACK window to mitigate the probability of ACK collision.

Our simulation has shown the CSD scheme recovers 85% channel throughput in networks under spurious CTS attacks as compared to regular networks. A JACK attacker can easily raise the energy consumption of a victim sender by 5 times and reduce the achievable throughput of the network to 15%. The ENAV scheme can recover a significant portion of the network throughput and reduces the energy consumption by the victim nodes to 40%.

We should emphasize that the essential difference between the SCTS and JACK: the former is a kind of virtual attack, and the latter is a physical attack.

In IEEE 802.11, the NAV is a virtual carrier sensing function. Through modifying victims' NAV value a SCTS attacker generate an illusion that the channel is keeping busy. Therefore it can get more opportunities to access unfair medium resource. To examine a CTS packet it is easy for nodes to assess the channel status through the physical carrier sensing function. It is an active process.

However, in the JACK attack, the spurious ACK packet is a physical signal. And the legitimate ACK packet is ruined physically by data collision. Therefore, avoiding from colliding with spurious ACK packet passively is the only way for victim to mitigate JACK attacks. Also the additional channel usage for extending the ACK transmission window lowers the network throughput. That is why ENAV scheme cannot recover wireless networks' throughput effectively (but work well on saving nodes' energy).

Choosing R dynamically can be adopted to increase the efficiency of CSD mechanism in the future work. Through assessing its no-ACK-packet loss level, nodes can adjust its optimal R. (For example, in a no ACK packet loss situation the R=1).

Appendix: NS2 tcl Script

NS2 is an opening powerful wireless networks simulation tool. It supports lots of popular protocols and maintains their integration. Tcl script processing as the interface provides necessary configuration information to the source codes, which is the core of NS2. Through a tcl script, researcher can tell the NS2 some critical factors that describe the profile the network: the number of nodes, their location, the topology, data rate, routing protocols, etc. Also, researcher can use tcl script to define each node' attribute such as link layer, interface queue, MAC layer etc. Or NS2 will simulate under default predefinition.

We provide the tcl script as a sample to illustrate the definition of the three-node model (cf. Figure 3.9).

```
# The simpliest test file of CSD scheme. Three nodes in a network. Malicious node B keeps sending out # CTS
packets. Node C tries to send something to node A.
#================================================================
set val(chan)           Channel/WirelessChannel          ;# channel type
set val(prop)           Propagation/TwoRayGround         ;# radio-propagation model
set val(netif)          Phy/WirelessPhy                  ;# network interface type
set val(mac)            Mac/802_11                       ;# MAC type
set val(ifq)            CMUPriQueue                      ;# interface queue type
set val(ll)             LL                           ;# link layer type
set val(ant)            Antenna/OmniAntenna              ;# antenna model
set val(x)              500                              ;# X dimension of topology
set val(y)              500                              ;# Y dimension of topology
set val(cp)             ""                           ;# node movement model file
set val(sc)             ""                               ;# traffic model file
set val(ifqlen)         5000                             ;# max packet in ifq
#set val(ifqlen)        50                               ;# max packet in ifq
set val(nn)             3                                ;# number of mobilenodes
set simCBRPktSize       1000                             ;# size of packet
set simCBRInterval      0.25
#set val(seed)          0.1
set val(stop)           12.0                             ;# simulation time
set val(tr)             3.tr                             ;# trace file name
set val(rp)             DSR                              ;# routing protocol
set AgentTrace          ON
set RouterTrace         OFF
set MacTrace            ON
```

```
# Initialize Global Variables
Mac/802_11 set bugFix_timer_ 1
Mac/802_11 set misrate_ 0.07
Mac/802_11 set nodenum_ $val(nn)
Mac/802_11 set scts_interval 0.045625
Mac/802_11 set rts_interval 45535
Mac/802_11 set spurious_cts_scheduledt_ 0
Mac/802_11 set malicious_0 0
Mac/802_11 set malicious_1 1
Mac/802_11 set zzg_debug 0
Mac/802_11 set spurNum_ 5
Mac/802_11 set dataRate_ 11Mb
Mac/802_11 set basicRate_ 11Mb
Phy/WirelessPhy set bandwidth_ 11Mb

set ns_                 [new Simulator]

if {$simSeed > 0} {
        puts "seed: $simSeed"
     ns-random $simSeed
}

# set up topography object
set topo [new Topography]

# Create God
create-god $val(nn)

#   Create the specified number of mobile nodes [$val(nn)] and "attach" them
#   to the channel. Three nodes are created : node(0), node(1) and node(2)
        $ns_ node-config -adhocRouting $val(rp) \
                         -llType $val(ll) \
                         -macType $val(mac) \
                         -ifqType $val(ifq) \
                         -ifqLen $val(ifqlen) \
                         -antType $val(ant) \
                         -propType $val(prop) \
                         -phyType $val(netif) \
                         -channelType $val(chan) \
                         -topoInstance $topo \
                         -agentTrace OFF \
                         -routerTrace OFF \
                         -macTrace ON \
                              -movementTrace OFF

#setup nodes
for {set i 0} {$i < $val(nn) } {incr i} {
      Mac/802_11 set iid [expr $i]
      set node_($i) [$ns_ node]
      $node_($i) random-motion 0
}

# set up the position of node0 (10.0, 20.0)
$node_(0) set X_ 10.0
$node_(0) set Y_ 10.0
$node_(0) set Z_ 0.0

#set up the position of node1 (150.0, 20.0)
$node_(1) set X_ 150.0
$node_(1) set Y_ 20.0
$node_(1) set Z_ 0.0

#set up the position of node1 (100.0, 200.0)
$node_(2) set X_ 150.0
$node_(2) set Y_ 10.0
$node_(2) set Z_ 0.0

#--------------------------------------------------------------
#link 1
# Setup traffic flow between nodes 0 connecting to 2 at time 2.0
set udp_(0) [new Agent/UDP]
$ns_ attach-agent $node_(0) $udp_(0)
set null_(0) [new Agent/Null]
$ns_ attach-agent $node_(2) $null_(0)

set cbr_(0) [new Application/Traffic/CBR]
$cbr_(0) set packetSize_ $simCBRPktSize
$cbr_(0) set rate_ 5Mb
$cbr_(0) set random_ 1

$cbr_(0) attach-agent $udp_(0)
$ns_ connect $udp_(0) $null_(0)
$ns_ at 2.0 "$cbr_(0) start"

#-------------------------------------------------
#Define node initial position in nam, only for nam
```

```
for {set i 0} {$i < $val(nn)} {incr i} {
    # The function must be called after mobility model is defined.
    $ns_ initial_node_pos $node_($i) 60
}

# Tell nodes when the simulation ends

for {set i 0} {$i < $val(nn) } {incr i} {
    $ns_ at $val(stop) "$node_($i) reset";
}

$ns_ at $val(stop)    "stop"
$ns_ at $val(stop)    "puts \"NS EXITING...\" ; $ns_ halt"

proc stop {} {
    global ns_ tracefd namfd
    $ns_ flush-trace
    close $tracefd
    close $namfd
    exit 0
}

puts "Starting Simulation..."
$ns_ run
```

Bibliography

[1] C. Siva Ram Murthy and B. S. Manoj. Ad Hoc Wireless Networks Architectures and Protocols. Professional Technical Reference, NJ 07458. Available at www.PHPTR.com.

[2] K. Liu, J. Deng, P. K. Varshney, and K. Balakrishnan, "An Acknowledgment-based Approach for the Detection of Routing Misbehavior in MANETs," IEEE Transactions on Mobile Computing, vol. 6, no. 5, pp. 536-550, May 2007.

[3] H. Yang, H. Luo, F. Ye, S. Lu, and L. Zhang, "Security in mobile ad hoc networks: challenges and solutions," in Wireless Communication, vol. 11, (UCLA, Los Angeles, CA, USA), pp. 38–47, Feb. 2004.

[4] P. Albers, O. Camp, J. Percher, B. Jouga, L. Me, and R. Puttini, "Security in ad hoc networks: a general intrusion detection," in Wireless Information Systems 2002, (Ciudad Real, Spain), pp. 1–12, Apr. 2002.

[5] L. Buttyan and J-P. Hubaux. Security and cooperation in wireless networks. Available at http://secowinet.epfl.ch.

[6] C. U. Saraydar, N. B. Mandayam, and D. J. Goodman, "Efficient power control via pricing in wireless data networks," *IEEE Transactions on Communications*, vol. 50, no. 2, pp. 291–303, February 2002.

[7] S. Radosavac, J.S. Baras and I. Koutsopoulos, "A framework for MAC layer misbehavior detection in wireless networks", Proceedings of ACM Workshop on Wireless Security (WiSe) 2005, Cologne, Germany.

[8] R. Negi and A. Rajeswaran, "DoS analysis of reservation based MAC protocols", IEEE International Conference on Communications, 2005.

[9] Gupta, V., Krishnamurthy, S.V., and Faloutsos, M., "Denial of Service Attacks at the MAC Layer in Wireless Ad Hoc Networks", Milcom 2002, Anaheim.

[10] Raj Jain. Wireless Local Area Networks (WLANs). Washington University Saint

Louis, MO 63130. Available at http:// www.cs.wustl.edu/~jain/cse574-06.

[11] IEEE 802.11 WG, ANSI/IEEE Std 802.11:*Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Medium Access Control (MAC) Enhancements for Quality of Service (QoS) IEEE 802.11/D2.0*, IEEE, 2001.

[12] L. Buttyan and J-P. Hubaux. Security and cooperation in wireless networks. Available at http://secowinet.epfl.ch.

[13] Giovanni Perbellini, "An Introduction to NS-2". Available at http: //esd.sci.univr.it.

[14] F. Stajano and R. Anderson, "The resurrecting duckling: Security issues for ad-hoc wireless networks," in Security Protocols: 7th International Workshop, (Cambridge, UK), pp. 172–182, Apr. 1999.

[15] H. P. Albers, O. Camp, J. Percher, B. Jouga, L. Me, and R. Puttini, "Security in ad hoc networks: a general intrusion detection," in *Wireless Information Systems 2002*, (Ciudad Real, Spain), pp. 1–12, Apr. 2002.

[16] S. Buchegger and B. Le, "Nodes bearing grudges: Towards routing security, fairness, and robustness in mobile ad hoc networks," in *Parallel, Distributed and Network-based Processing, 2002*, (Canary Islands, Spain), pp. 403–410, 2002.

[17] D. Djenouri, L. Khelladi, and A. Badache, "A survey of security issues in mobile ad hoc and sensor networks" in *Communications Surveys and Tutorials*, vol. 7, pp. 2–28, Fourth Quarter 2005.

[18] The network simulator (ns-2). http://www.isi.edu/nsnam/ns/.

[19] IEEE Computer Society, "Wireless LAN medium access control (MAC) and physical layer (PHY) specifications," IEEE standard 802.11, 1999.

[20] J Parker, A Patwardhan, A Joshi, "Detecting wireless misbehavior through cross-layer analysis", Proceedings, IEEE Consumer Communications and Networking Conference Special Sessions, January 2006.

[21] Mithun Acharya, David Thuente "Intelligent Jamming Attacks, Counterattacks and

(Counter) ^2 attacks in 802.11b Wireless Networks”, OPNETWORK-2005 Conference, Washington DC,August 2005.

[22] D. Chen, J. Deng , and P. K. Varshney, "Protecting Wireless Networks against a Denial of Service Attack Based on Virtual Jamming," ACM MobiCom '03, Poster, San Diego, CA, USA, September 14-19, 2003.

[23] M. Takai, J. Martin, and R. Bagrodia, "Effects of Wireless Physical Layer Modeling in Mobile Ad Hoc Networks", In Proceedings of MobiHoc 2001, pages 87-94, October 2001.

[24] John Bellardo and Stefan Savage, "802.11 Denial-of-Service Attacks: Real Vulnerabilities and Practical Solutions", In Proceedings of the USENIX Security Symposium, Washington D.C., August 2003.

[25] W.A. Arbaugh, N. Shankar, J.Wang, and K. Zhang. Your 802.11 Network has No Clothes. In *First IEEE International Conference on Wireless LANs and Home Networks*, Suntec City, Singapore, December, 2001.

[26] Mike Lynn and Robert Baird. Advanced 802.11 Attack. Black Hat Briefings, July 2002.

[27] Daniel B. Faria and David R. Cheriton DoS and Authentication in Wireless Public Access Networks. In *Proceedings of the First ACM Workshop on Wireless Security (WiSe’02)*, September 2002.

[28] Saikat Ray, Jeffrey B. Carruthers and David Starobinski, "RTS/CTS-Induced Congestion in Ad Hoc Wireless LANs," WCNC 2003: Wireless Communications and Networking Conference: New Orleans March 16 - 20, 2003.

[29] Pradeep Kyasanur and Nitin Vaidya, "Detection and Handling of MAC Layer Misbehavior in Wireless Networks", Dependable Computing and Communications Symposium (DCC) at the International Conference on Dependable Systems and Networks (DSN), June 2003.

[30] Alvaro Cardenas, Svetlana Radosavac and John S. Baras, "Detection and Prevention

of MAC Layer Misbehavior for Ad Hoc Networks", 2004 ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN 2004), pages 17-22, Washington DC, USA.

[31] S.-R. Ye, Y.-C. Wang and Y.-C. Tseng, "A Jamming-Based MAC Protocol to Improve the Performance of Wireless Multihop Ad Hoc Networks", Wireless Communications and Mobile Computing, Vol. 4, No. 1, Feb. 2004, pp. 75-84. (SCIE).

[32] M.Cagalj, S.Ganeriwal, I. Aad, J. P.hubaux, "On cheating in CSMA/CA ad-hoc networks," IEEE INFOCOM 2005.

[33] S.Xu and T.Saadawi," Does the IEEE 802.11 MAC protocol work well in multihop wireless adhoc networks", IEEE Communications Magazine, Volume. 39, Issue 6, Jun 2001.

[34] S. Radosavac, J.S. Baras and I. Koutsopoulos, "A framework for MAC layer misbehavior detection in wireless networks", Proceedings of ACM Workshop on Wireless Security (WiSe) 2005, Cologne, Germany.

[35] Design Challenges for Energy-Constrained Ad Hoc Wireless Networks. Goldsmith, Stephen B. Wicker. IEEE Wireless Communications, August 2002.

[36] L. M. Feeney and M. Nilsson. Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In *IEEE INFOCOM*, 2001.

[37] L. Zhou and Z. Haas. Securing ad hoc networks. IEEE Network 13(6):24--30, November/December 1999.

[38] Computer Networks (Fourth Edition), Author Andrew S.Tanenbaum, Publisher Prentice Hall PTR, ISBN 0130661023.

[39] The ns Manual -- A collaboratoin between researchers at UC Berkeley, LBL, USC/ISI, and Xerox PARC. Editor Kevin Fall, Editor Kannan Varadhan

[40] IEEE. IEEE wireless LAN medium access control (MAC) and physical layer (PHY) specifications, 1999.

[41] M. Raya, J.-P. Hubaux, and I. Aad, “DOMINO: A system to detect greedy behavior in IEEE 802.11 hotspots,” in ACM MobiSys, June 2004.

[42] R. Negi and A. Perrig, “Jamming Analysis of MAC protocols,” Tech. Memo Carnegie Mellon Univ., Feb, 2003.

[43] J. Deng, Z. Zhang, S. Pagadala, and P. K. Varshney, "Protecting MANETs from Spurious CTS Attacks with Randomized Carrier Sensing," in Proc. of IEEE Sarnoff Symposium '08, Princeton, NJ, USA, April 28-30, 2008.

[44] US Berkley, “The NS Manual”, Kevin Fall Editor, Kannan Varadhan Editor.

[45] Vaduvur Bharghavan, Alan J. Demers Scott Shenker, and Lixia Zhang. MACAW: A Media Access Protocol for Wireless LAN’s. In *Proceedings of the ACM SIGCOMM Conference*, London, UK, September 1994.

[46] Nikita Borisov, Ian Goldberg, and David Wagner. Intercepting Mobile Communication: The Insecurity of 802.11. In *Seventh Annual International Conference on Mobile Computing And Networking*, Rome, Italy, July 2001.

Vita

Zhiguo Zhang was born in Hubei province, P.R.China. He is the second child in the family, and has two siblings. He received his B.S. degree and M.S. degree in Mechanical Engineering from Harbin Institute of Technology in Harbin, in 1997 and 1999 respectively. Then he worked in Time Group Inc, Beijing as an engineer from Jul. 1999 to Feb. 2001. In 2001 February, he moved to Nanjing to work in ZTE Corporation as a senior software engineer from Feb. 2001 to Feb. 2005.He has started his study for Master's Degree in Computer Science department of University of New Orleans since 2006. His research focuses on mobile ad hoc networks. His email is zzhang6@uno.edu.