ALIBI FRAMEWORK FOR IDENTIFYING INSIDER JAMMING ATTACKS IN HALF-DUPLEX WIRELESS LOCAL AREA NETWORKS

BY

HOANG VIET NGUYEN

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2010

Urbana, Illinois

Doctoral Committee:

Professor Klara Nahrstedt, Chair
Professor Roy Campbell
Professor Nitin Vaidya
Associate Professor David Yau, Purdue University

ABSTRACT

Recent advances in wireless communications and digital electronics have enabled rapid development of a variety of wireless network technologies, such as wireless LANs, home networks, multi-hop ad hoc networks, and sensor networks. Wireless networks, unfortunately, are vulnerable to radio jamming attacks (in short, "jamming attacks") due to the open and shared nature of wireless medium. In a jamming attack, an attacker injects a high level of noise into the wireless system which significantly reduces the signal-to-noise ratio (SINR) and reducing the probability of successful message receptions. Even though the spread spectrum technologies have raised the bar for the jamming defenses, they cannot deal with insider jammers who launch the stealthy and intelligent jamming attacks from compromised nodes. To cope with such dangerous insider jammers, the first and most important step is to *identify* them.

In this dissertation, we consider the problem of identifying the insider jammers. Our approach to this problem is unique: we exploit the half-duplex nature of the attackers. Specifically, a half-duplex jammer has the following characteristics:

- It cannot *send* on two different channels simultaneously due to a non-negligible channel switching time.
- It cannot *receive* on two different channels simultaneously due to a non-negligible channel switching time.
- It cannot *send* and *receive* on a channel simultaneously due to a non-negligible transmit-to-receive switching time.

Therefore, when a compromised node jams, it cannot either send or receive any other packets. More importantly, if an honest node is observed doing a send or receive action at the same time of the jammed packet, it can arguably

prove that it cannot be the cause of the jammed packet. In other words, the honest node obtains an "alibi".

Alibi is "a form of defense whereby a defendant attempts to prove that he or she was elsewhere when the crime in question was committed". In the context of jamming attacks, an alibi for a node is a proof showing that an honest node could not commit a jamming action at a specific time because it was witnessed doing a legitimate action at the same time. We focus on exploring the alibi framework in dealing with insider jammers. We study various properties of the framework including detection accuracy, detection time, network availability and necessary conditions for the alibi framework to work. We also investigate different designs of the alibi framework such as sending-based alibis and receiving-based alibis and study their strengths and weaknesses. We evaluate the alibi framework by the analysis, simulations and MICAz experiments. To the best of our knowledge, the alibi framework is the first framework exploiting the half-duplex nature of the nodes to identify insider attackers.

To my parents, Phien Nguyen and Thao Mai
and my dear wife, Ngu Truong.

ACKNOWLEDGMENTS

First and foremost, I would like to express my deepest gratitude to my advisor, Professor Klara Nahrstedt, for her invaluable guidance and continuous support throughout my PhD research. Her wide knowledge and key insights have been a constant help during my pursuit of research. Her patient advice and thoughtful directions have greatly improved my research, writing and presentation skills. To me, she has been a great advisor and an irreplaceable role model. I will always be grateful for her consistent encouragement and belief in me during my periods of doubt and frustration. I feel extremely lucky to have Klara as my advisor, who is always dedicated to the success of her students.

I would like to thank Professor Roy Campbell for his advice on the early ideas and insightful feedbacks on the thesis. The discussions with him have always brought many practical aspects into the thesis. My sincere gratitude goes to Professor Nitin Vaidya who not only has given a lot of encouragement but also a lot of insightful comments on the thesis. I am also thrilled to have the backing of Professor David Yau for this dissertation. I really appreciate all the members of my dissertation committee, Professor Klara Nahrstedt, Professor Nitin Vaidya, Professor Roy Campbell, and Professor David Yau, for their invaluable comments and constructive suggestions to improve the work. I hope to continue this collegial relationship with my committee members.

The members of MONET (Multimedia Operating System and Networking) group have given me a supportive and comfortable environment to work in. My special thanks go to Thadpong Pongthawornkamol, who has spent countless number of hours giving feedbacks on the development of the alibi framework. I also would like to thank William Conner, Long Vu, Ying Huang, Wanmin Wu, Rahul Malik, Raoul Rivas, Qiyan Wang, Ahsan Arefin, Shameem Ahmed, Debessay Fesehaye and Zixia Huang. Anda Ohlsson,

Lynette Lubben, Mary Beth Kelley and Amanda Brown deserve a special mention for efforts to ensure that all administrative procedures go through smoothly.

I also would like to thank all friends I have met and worked with during my PhD study. Special thanks go to Jay Patel for cheering me up when I almost gave up on the PhD. I also would like to thank Loc Bui, Duan Tran, Du Tran, Tung Le, Kien Chi and many other Vietnamese friends who have made my life at UIUC fun and relaxing.

During this course of work, at University of Illinois, I was supported by Vietnam Education Foundation from 2004-2005 and by NSF under the grant CNS-0524695, ITI (Information Trust Institute) and TCIP center (Trustworthy Cyber-Infrastructure for the Power Grid) from 2006-2010.

I would like to thank my entire family in Vietnam for their understanding and continuous support during these years. At times, it has been good to know they have just been a phone call or an email away.

Last but not least, a big thank goes to my wife, Ngu Truong, for her unconditional love, supports and encouragement, for her patience to listen to my endless complaints and frustrations about PhD life, for cheering me up when I was down, for reminding me to stay focused on research, for giving me so many joyful and happy moments and for always believing in me. Without her, it would have been certainly much harder to finish my PhD.

This thesis is dedicated to my family and my wife.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Recent advances in wireless communications and digital electronics have enabled rapid development of a variety of wireless network technologies, such as wireless LANs, home networks, multi-hop ad hoc networks, and sensor networks. Wireless networks, unfortunately, are vulnerable to radio jamming attacks (in short, "jamming attacks") due to the open and shared nature of wireless medium. In a jamming attack, an attacker injects a high level of noise into the wireless system which significantly reduces the signal-to-noise ratio (SINR) and the probability of successful message receptions.

Jamming attacks are serious in several ways. First, the jamming attack is a type of Denial-of-Service attacks (DoS). Jammed communication channels are useless most of the time. Second, it is relatively easy to perform a jamming attack. The attacker only needs a transmitter (i.e. jamming device) powerful enough to transmit a signal to disrupt the targeted wireless communication because the wireless medium is open and shared in nature. For example, an inexpensive device able to transmit signal on 2.4Ghz is enough to jam a 802.11b network [1, 2]. Third, it is hard to detect the jamming attack (i.e. the existence of the attack) and identify/locate the attacker. The main reason is due to the ambiguity between unintentional interference and intentional jamming attacks [3]. Lastly, even if the jamming attack and the attacker are detected, it is very challenging to automatically recover from the jamming attacks [4, 5, 6]. The network needs an out-of-band means to defense against the attacks (e.g. having a person remove the jamming device or having the network do a spatial retreat [4]).

There are two types of jamming strategies in wireless networks [3]: *proactive* and *reactive* jamming strategies. In proactive jamming strategies, attackers jam channels regardless of whether there are on-going communication activities on the channels. Typical examples of proactive jamming strategy are continuous jamming, periodic jamming and random jamming strategy.

The proactive jamming strategies are easy to implement but they are not energy efficient.

In the reactive jamming attacks, in contrast, the attackers only jam the channels when there are on-going communication activities. Typical examples are scan-and-jam (on multi-channel networks) and listen-and-jam strategy (on single-channel networks). The difficulty in defending against reactive jamming strategy lies on attackers' ability to become stealthy. For example, in a listen-and-jam strategy in a wireless LAN (WLAN), an attacker looks for an preamble signal and immediately sends a short packet to corrupt the body of the incoming packet. To become stealthy, it may want to jam some (possibly important) packets and behave like other nodes for the rest of the time. In this way, besides the fact that the attacker can do significant damage to the network, it also makes other honest nodes to back off more frequently and can enjoy extra throughput when behaving honestly.

One of the most effective jamming mitigation is the *spread spectrum* technique. By hopping the carrier frequency or spreading the signal in time, the wireless network can force the jammer to spend several-fold more power than if the spread spectrum were not used [7]. In the classic spread spectrum approach, the knowledge of the spreading factor such as the hopping pattern and the pseudo-noise chip is a *common* knowledge that can be obtained easily [8]. With such a knowledge, the attackers can jam the networks using the classic spread spectrum (e.g. IEEE 802.11 [8]) very easily [2]. Therefore, there is a strong need to make such common knowledge *secret*.

An obvious approach is to make the common knowledge as *shared* secrets among nodes. This will force the *outsider* jammer, having no knowledge of the shared secrets, to spend much more power than if the jammer knew the shared secret[9, 10, 11]. However, this approach does not work with *insider* jammers who have the knowledge of the shared secrets (e.g. via compromised nodes in the network). Thus, there are stronger security mechanisms needed to deal with insider jammers.

One approach to deal with insider jammers is to use multiple shared secrets rather than just one single shared secret. In [12, 5, 13], the common idea is to divide nodes into multiple groups and assign one unique shared secret to each group. The assigned shared secret of a group is used to derive the hopping-pattern for that group to the base station. If the communication of a group is jammed, one group member must be a jammer because no nodes

outside the group know the shared secret. By further dividing the jammed groups into sub-groups with different shared secrets, the network can avoid mitigate the jamming effect caused by the compromised shared secrets.

The other approach to deal with insider jammers is to remove the dependencies on the pre-shared secret of traditional spread spectrum technologies. Recently, researchers have been proposing spread spectrum communication without any pre-shared secrets. Uncoordinated Frequency Hopping (UFH) [14, 15, 16] allows two nodes that do not have any common secret to establish a secret key for future FHSS communication. Uncoordinated Direct Sequence Spread Spectrum (UDSSS) [17] avoids jamming by randomly selecting a spread code sequence from a pool of code sequences. However, UDSSS is vulnerable to reactive jamming attacks. RD-DSSS [18] also proposes a similar technique that can be resistant to reactive jamming attacks. BBC [19] proposes a coding approach to have jamming-resistant communication without any prior knowledge of keys.

The major problem with the multi shared secrets and zero-shared secrets is that either 1) the network has much lower throughput compared to the single-shared secret approach, or 2) the network needs a special base station that can send/receive simultaneously in all channels. Therefore, these approaches should be used to build *alternative* communications under the jamming situations. To this end, it is still most preferable to *identify* and *eliminate* the insider jammers.

1.1 Problem Description

In this dissertation, we study the problem of *identifying* insider jamming attacks in half-duplex wireless LAN. Specifically, in an insider jamming attack, several nodes are compromised and become the source of jamming. The goal of the attackers is to remain undetected while maximizing possible damages by jamming actions.

There is a fundamental difference between jamming *identification* problem and jamming *detection* problem. The latter is concerned about *how to detect the presence of a jamming attack* while the former is concerned about *how to identify which nodes are the jammers*. Note that we are only concerned with attacker identification. It is trivial that once the insider jammers

are identified, they can be easily removed from the network by applying physical responses (e.g. node shutdown) or by delivering new shared secrets to the honest nodes.

The problem of identifying insider jammers is very challenging due to the following reasons.

- *Stealthy attacks*: The insider attackers can act stealthily to hide themselves in the system. They are assumed to use reactive jamming strategies to jam packets. That means, they only jam when there is on-going communication.
- *Inside knowledge of the networks*: The attackers are assumed to know any inside knowledge of the network such as shared secrets and network protocols.
- *Multiple attackers*: There might be more than one attacker in the system. These attackers may or may not collude with each other.
- *Collision vs. Jamming actions*: There are no ways to differentiate collisions and jamming actions because jamming actions can be considered as *intentional collisions*.

1.2 Our Approach

Our approach to identifying insider jammers is novel and unique: we exploit the half-duplex nature of the jammers. Specifically, a half-duplex jammer has the following weaknesses:

- It cannot *send* on two different channels simultaneously due to a non-negligible channel switching time.
- It cannot *receive* on two different channels simultaneously due to a non-negligible channel switching time.
- It cannot *send* and *receive* on the same channel simultaneously due to a non-negligible transmit-to-receive switching time.

We propose a novel concept called "alibi". Alibi is "a form of defense whereby a defendant attempts to prove that he or she was elsewhere when

the crime in question was committed". In the context of jamming attacks, an alibi for a node is a proof showing that the node could not commit a jamming action at a specific time because it was witnessed doing a legitimate action at the same time. Legitimate actions can be "transmitting at a specific channel" or "receiving a packet". An important property of alibi is that an attacker will never be able to obtain an alibi by itself when it jams. Thus, for a certain time period, an attacker who jams at a certain jamming rate cannot obtain as many alibis as other nodes. To this end, it is possible to identify the attackers in the network based on the rate at which nodes obtain alibis.

The design of the alibi framework depends on several important factors.

- *Alibi definition*: If alibis are based on legitimate sending actions of honest nodes, referred to as *sending-based alibis*, we have a sending-based alibi framework. If alibis are based on legitimate receiving actions of honest nodes, referred to as *receiving-based alibis*, we have a receiving-based alibi framework.

- *Network characteristics*: Several network characteristics have important aspects on the design of alibi framework: single-/multi-channel capability, delay characteristics of nodes including channel switching delay and transmit-to-receive delay.

1.3 Contribution and Dissertation Outline

In this dissertation, we focus on exploring the alibi framework in dealing with insider jammers. We study various properties of the framework including detection accuracy, detection time and the network availability and the necessary conditions for the alibi framework to work. We also investigate different designs of the alibi framework such as sending-based alibis and receiving-based alibis and study their strengths and weaknesses. To the best of our knowledge, the alibi framework is the first framework exploiting the half-duplex nature of the nodes to identify insider attackers.

We will explore different designs of the alibi framework in this dissertation. The ontology of the alibi framework is shown in Figure 1.1. The alibi framework can be designed using sending-based alibis (S-alibi) or receiving-based alibis (R-alibi). For S-alibis where alibis are defined over multiple

channels, the S-alibi scheme can only be defined on multi-channel networks only. S-alibi scheme can also deal with non-colluding and colluding attackers. For R-alibis where the alibis are defined at the reception of nodes, it is possible to design the R-alibi schemes on both single- and multi-channel that can cope with non-colluding and colluding attackers.

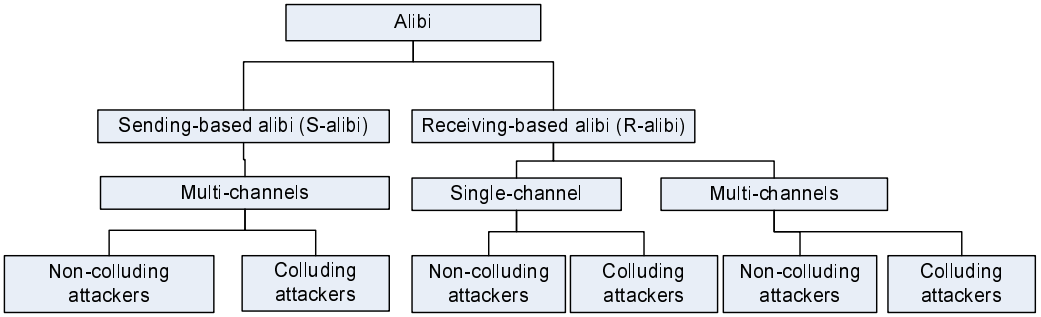


Figure 1.1: Alibi Ontology

The dissertation is organized as follows. In Chapter 2, we give common models and assumptions in the thesis. In Chapter 3, we give a generalized alibi framework with several important definitions, the principle of operation, architectures, desired properties and some common techniques of any alibi framework. In Chapter 4, we present the alibi design for multi-channel wireless LANs in which both sending-based and receiving-based alibis are used. Then, we present the receiving-based alibi design for single-hop ad hoc networks in Chapter 5. In Chapter 6, we further extend the receiving-based alibi design for single-channel wireless LANs. Chapter 7 presents related work to this dissertation. We conclude the dissertation and discuss the future directions in Chapter 8.

CHAPTER 2

MODELS AND ASSUMPTIONS

In this chapter, we give common assumptions, notations and definitions used throughout in the thesis. Each chapter, however, may have additional assumptions, notations and definitions.

2.1 Network Model

We consider a wireless LAN where each node can communicate within one hop with a trusted base station BS. Denote $\mathcal{N} = 1, \ldots, |\mathcal{N}|$ the set of nodes in the network. Denote $n = |\mathcal{N}|$ the number of nodes in the network. Each node i in the network is equipped with *one* half-duplex radio. That means,

- It cannot *send* on two different channels simultaneously due to a non-negligible channel switching time τ_{cs}.
- It cannot *receive* on two different channels simultaneously due to a non-negligible channel switching time τ_{cs}.
- It cannot *send and receive* on a channel simultaneously due to a non-negligible transmit-to-receive switching time τ_{txrx}.

We assume CRC-failed packets are still delivered to the upper layer. Each node may have a set of (physical) channels $\mathcal{C} = \{c_1, c_2, \cdots, c_{|\mathcal{C}|}\}$ to switch to. In a single-channel network, $|\mathcal{C}| = 1$. In a multi-channel network, $|\mathcal{C}| > 1$.

The base station BS and the nodes in $\mathcal{N}$ communicate over a *control channel* to exchange control information. In single-channel networks, the control channel uses the same physical channel as the data channel. In multi-channel networks, the control channel can be either a dedicated physical channel or can be constructed via multiple (physical) channels. The

required feature of the control channel is that it has *non-zero throughput under the jamming situation*. The specific design of the control channel will be discussed later in each alibi scheme.

We assume honest nodes cooperate and follow the protocol. The attackers, however, can deviate arbitrarily from the protocols.

2.2 Attack model

We assume some nodes in the network are the attackers. Denote $\mathcal{J} = \{j_1, j_2, ..., j_{|\mathcal{J}|}\}$ the set of the jammers in the network ($\mathcal{J} \subset \mathcal{N}$). Therefore, the attackers have the same physical capabilities as other nodes. However, the attackers have the complete control of the MAC, physical parameters of the radio network interface. We particularly focus on the jamming attacks done through the malicious modification of the MAC protocol and parameters. Specifically, the attackers jam the network by *intentionally* sending jamming packets on the target channels. The desirable result of a jamming packet is to collide with a legitimate packet sent by an honest node, resulting in a corrupted and un-decodable packet. From the MAC point of view, this type of jamming attacks may be considered as *intentional* collisions. In contrast, collisions in wireless networks are *unintentional*.

The attackers are insider attackers. They are assumed to know any security-related information of the node such as security keys. They also know any shared secret and protocols used in the system. Therefore, the design of the alibi framework should not rely on any shared secrets in the network.

The attacker is assumed to use a reactive jamming strategy due to the effectiveness of the strategy in terms of energy and stealthiness. That means, it only jams when it knows that there is an on-going packet. There are several ways to implement this strategy. For CSMA/CA network where the channel access is essentially random, every packet transmission has to start with a *preamble* to inform the intended receiver and other nodes in the network about the incoming packet. Following the preamble is the body of the packet including the MAC layer information and other application information. By first sensing a preamble and immediately transmitting a short packet, an attacker can successfully jam a packet in a reactive way.

Figure 2.1 illustrates a reactive jamming attack on the CSMA/CA network. The same jamming strategy can definitely be applied to any other networks as long as the attackers have the capability to sense activities on the target channel. In case the radio of the nodes and the attackers cannot sense channel activities, nodes will have to rely on TDMA scheduling. In such a case, the attackers need to jam the targeted slots because they know there should be communications during the slots.

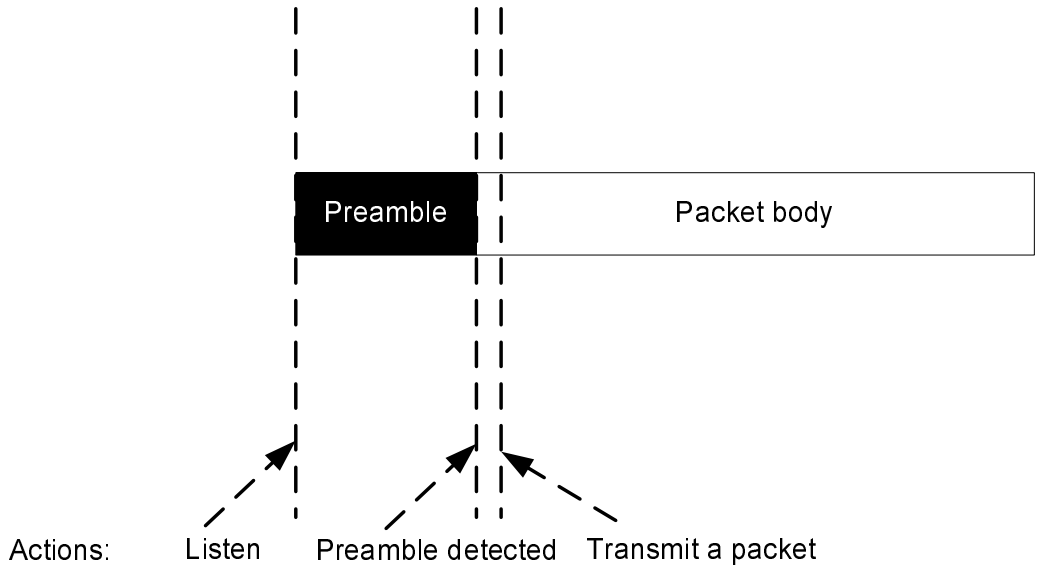


Figure 2.1: Reactive jamming attacks on CSMA/CA

The goal of the attackers is to remain undetected while maximizing the number of jammed packets. The attackers use probabilistic reactive jamming strategy. That means whenever an attacker j senses an on-going packet by detecting the presence of a preamble, it will transmit a jamming packet with probability p_j. p_j is called the "reactive jamming probability" and is defined for *each sending packet*. This definition is different from traditional jamming rate in the literature which is defined over *each time slot* regardless of whether there is sending packet in that time slot. Henceforth, the term "jamming rate" refers to "reactive jamming rate" unless explicitly specified.

2.3 Detection Model

We assume a central detection model. The central detector is placed on the trusted node BS. Nodes will send information to the trusted node BS. Each

node assumes to have a shared key with the central detector BS so that it can have a secret communication with BS.

CHAPTER 3

GENERALIZED ALIBI FRAMEWORK

In this chapter, we present the general alibi framework including definitions, notations and desired properties. Essentially, this is an abstract framework and is applicable for any instance of the alibi framework.

3.1 Definitions

Alibi is "a form of defense whereby a defendant attempts to prove that he or she was elsewhere when the crime in question was committed". In the context of jamming attacks, an alibi for a node is a proof showing that the node could not commit a jamming action at a specific time because it was witnessed doing a legitimate action at the same time. Legitimate actions can be "transmitting at a specific channel" or "receiving a packet". An important property of alibi is that an attacker cannot get an alibi by itself when it jams due to the limitation of the half-duplex radio.

In the alibi framework, each node may play two roles: *defendant role* and *witness role*. A node takes on the defendant role when it does legitimate actions to prove its honesty. The result of a defending action is a *proof* held by witnesses. If proofs are combined properly, the defendant gets an *alibi*. A node voluntarily plays the witness role when it *observes* a legitimate action done by a defendant.

There are two types of alibis: sending-based alibis (S-alibis) and receiving-based alibis (R-alibis). S-alibis rely on the sending actions of the defendant while R-alibis rely on the receiving action of the defendant.

Definition 1 (Sending-based alibi (S-alibi)) *A sending-based alibi for a node shows that the defendant was observed, by several witnesses, sending an uncorrupted packet over one time slot in one channel at the time the jamming action took place in another channel, also observed by several other witnesses.*

S-alibi has the following advantages and disadvantages.

- *Uniqueness*: there can only be one node claiming a S-alibi in a channel at a time slot. This property limits the possibility of sharing the S-alibis among colluding attackers.
- *Multi-channel*: S-alibi is only possible in multi-channel networks because it requires sendind actions in different channels.

Definition 2 (Receiving-based alibi (R-alibi)) *A receiving-based alibi for a node shows that the defendant was receiving a packet at the time the jamming action took place.*

R-alibi has the following advantages and disadvantages.

- *Single/Multi-channel*: Besides the possibility on multi-channel networks like S-alibis, R-alibis can also be possible on single-channel networks. This is due to the possibility of multiple receptions of the packet on a single channel.
- *Transferability*: An R-alibi for a node can be transferred to other nodes. This property makes R-alibis weak against colluding attackers who may want to share R-alibis among themselves.

3.2 Operation Principle of Alibis

The key principle in using alibis to identify insider jammers is that there has to be significant difference of alibis obtained by honest nodes and attackers. The difference can be deterministic such as "only good nodes can obtain alibi while attackers cannot" or statistical such as "a good node statistically obtains higher number of alibis than an attacker". With these differences, as time goes on, the attackers will be eventually identified. However, if the attackers can manage to remove the differences, the alibi framework will fail to differentiate the good nodes and the attackers. Thus, it is very important to have the right definition and implementation of alibis.

3.3 Alibi Architecture

The architectures of the generalized alibi framework for the base station BS and the clients $i \in \mathcal{N}$ are shown in Figure 3.1 and Figure 3.2, respectively. Note that the architectures only show the components relevant to the alibi framework.

MAC is a component for nodes to access the shared medium. Depending on the type of the networks, the MAC can be CSMA/CA or TDMA. The characteristics of the MAC component will heavily influence the design of the BBC-based control channel.

Monitor is a component lying above the MAC layer to monitor the channel. It specifically looks for corrupted packets and pass them to the upper layer with the corresponding timestamps to create potential proofs. For a client, the Monitor puts proofs into a finite queue. For the base station, the Monitor passes proofs to the Proof Collector.

Alibi protocol at the clients is designed to receive proofs and send them to base station.

Jamming-resistant control channel is the main communication in the jamming situation and is built on top of the MAC component. The specific design of the jamming-resistant control channel will be discussed further in each instance of the framework. The common characteristic of the jamming-resistant control channel is that it relies on no pre-shared secrets between communication parties and thus has low throughput. It should be used to send control information only.

Proof Collector is the component at the base station BS only. It collects proofs from different sources, combines them and generates alibis for nodes. It also may issue the "collect-proof" command through the jamming-resistant control channel to collect proofs from the nodes.

Alibi detector is the component at the base station only. It receives alibis of each node from the Proof Collector and performs the detection algorithm.

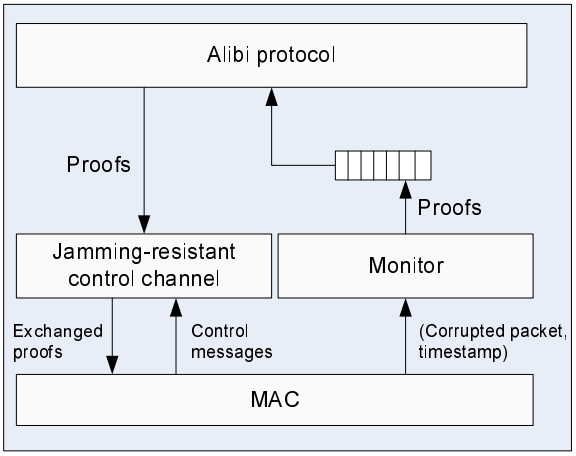


Figure 3.1: Alibi architecture of nodes

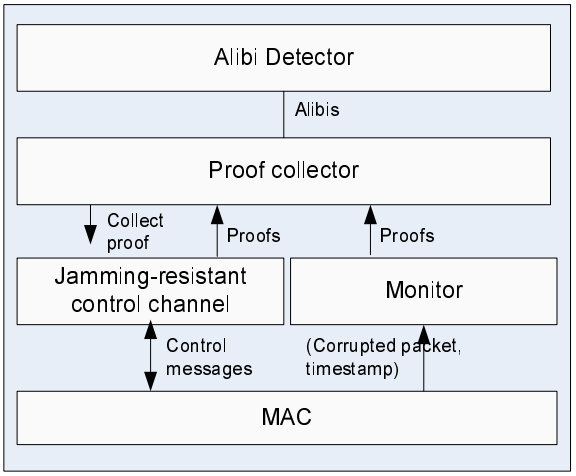


Figure 3.2: Alibi architecture of the base station

3.4 Desired Properties

An alibi framework needs to have five properties: completeness, accuracy, detection time, availability and scalability.

- *Accuracy*: This property is concerned about the false alarm rate and the detection probability. A false alarm is declared when an honest node is falsely accused as an attacker. The detection probability is defined as the number of true positives over the total number of identifications.
- *Detection time*: This property is concerned about the time to identify the attackers. Specifically, any alibi-based detection algorithms must show that the time to detect is bounded.
- *Availability*: This property is defined as the fraction of time network can successful communication. Intuitively, the more the attackers jam the channels, the less availability is and the faster the attackers get detected. Thus, this becomes the trade-off for the attackers.
- *Scalability*: This property specifies how much overhead is incurred in terms of storage, computation and communication overhead.

3.5 Jamming-resistant Control Channels

There are several ways to create jamming-resistant control channel on top of the MAC layer. In this section, we present two designs of jamming-resistant control channels. One design is based on concurrent code (BBC code) [19] which can work on single-/multi-channel networks. The other design relies on the uncoordinated frequency hopping (UFH) proposed in [15]. The main reason for using these schemes is that they can work on the MAC layer without any modifications of the physical layer. There are other ways to design jamming-resistant control channels such as UDSSS [17] and RD-DSSS [18] that require certain modifications in the physical layer.

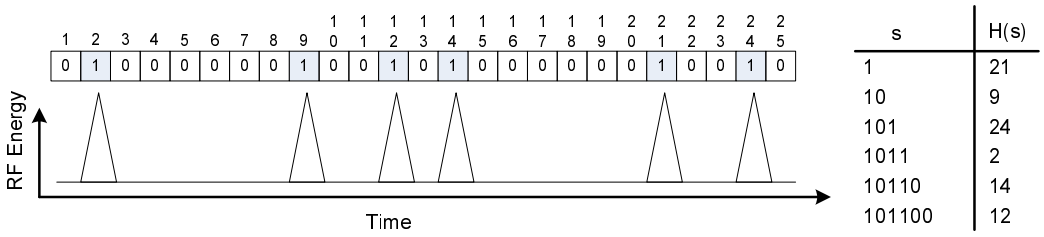


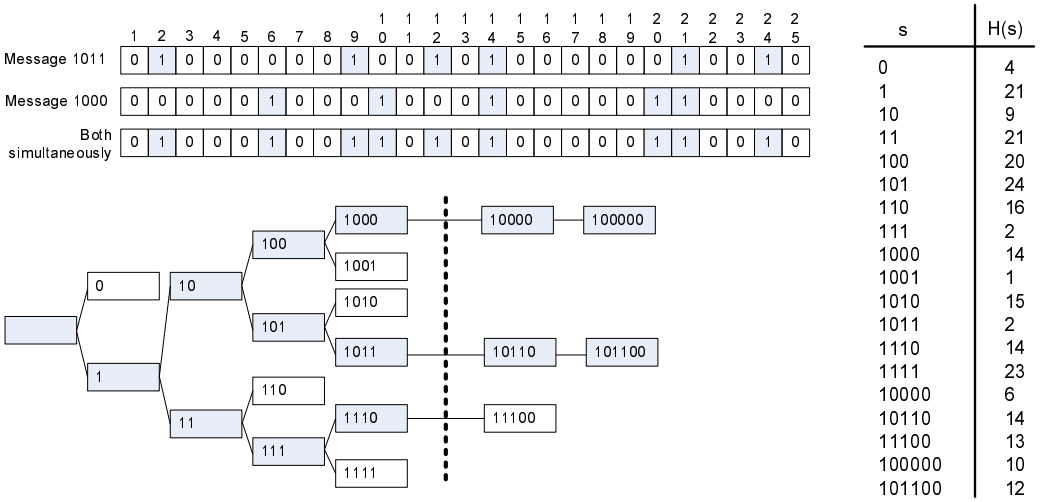
Figure 3.3: An example of BBC broadcast using pulses (excerpted from [19]).

3.5.1 Jamming-resistant Control Channel using Concurrent Code (BBC code)

BBC is a keyless jamming-resistant broadcast communication proposed in [19]. The basic idea is to have the sender create *"indelible" marks* in an additive-OR channel to convey a sending message. The receiver receives a "packet" containing all the marks and decodes the original sending message. Indelible marks in the additive-OR channel have an important property: the jammers cannot erase the existence of the marks in the channel; they can only create extra marks. Therefore, a received BBC packet may contain more marks than those created by the senders. In other words, the set of marks in the received BBC packet is the super-set of the marks created by senders. Thus, the coding scheme is called *concurrent code* and BBC code is the only known implementation of the concurrent code.

A message M of length n bits is encoded into a message M' of length $m = ne$, where e is the expansion factor ($e > 1$). Denote $\Pi(m, i)$ to be the first i bits of message m and a hash function H (e.g. MD5 or SHA1). For each $i \in (1, n)$, we calculate the location $L(i)$ of the i-th mark corresponding to bit i as $L(i) = H(\Pi(m, i)) \mod m$. Therefore, the message M is encoded into n marks whose locations are from $0..m$. The encoded message M' will then be transmitted in m slots. The marked slots correspond to transmissions of a pulse (i.e. a preamble-only packet). The unmarked slot is equivalent to no transmission. Figure 3.3 shows an example of a BBC encode and broadcast using pulses. Message $M = 1011$ is padded with two 0-bits and becomes $M = 101100$. An *indelible mark* is a radio pulse, and its location is the time when pulse occurs relative to the start of the message. The table on the right shows part of the definition of the hash function $H(x)$. Each prefix of the padded message is hashed using the hash function $H(x)$. The results of the

hash are used as the locations of the pulse.



s	H(s)
0	4
1	21
10	9
11	21
100	20
101	24
110	16
111	2
1000	14
1001	1
1010	15
1011	2
1110	14
1111	23
10000	6
10110	14
11100	13
100000	10
101100	12

Figure 3.4: Decode tree for the BBC broadcast using pulses (excerpted from [19]).

The receiver receives a packet of length m containing at least n marks. The packet may have more than n marks due to either multiple messages sent or extra marks by jammers. The decoding process guarantees to decode any messages contained in the packet. It maintains a tree of maximum depth n rooted by an empty message. At the depth level 1, if the location at ($H(0)$ mod m) is marked, 0 is added into the tree. Similarly, if the location of ($H(1)$ mod m) is marked, 1 is added into the tree. Recursively, at each node at level i with the corresponding message of length i M_i, if ($H(M_ib)$ mod m) ($b = 0, 1$) is marked in the packet, b is added as a child of the node. The tree keeps growing until the level n. Any nodes at level n correspond to decoded messages of length n. Figure 3.4 shows an example of the decode tree for the BBC broadcast using pulses. In this example, two messages 1011 and 1000, padded with two 0-bits for the checksum purpose, were broadcasted simultaneously. The table on the right shows part of the definition of the hash function $H(x)$. The 3rd row at the top shows at which times a pulse was sent (gray boxes with 1s) or not sent (white boxes with 0s). This is essentially a bit-wise OR of the pulse pattern for each message. The binary tree shows those prefixes that are considered during decoding. A prefix is gray if its hash gives a time at which a pulse was detected. The dotted line shows the transition from decoding the body of the message to checking the appended checksum zeros.

Even though the decoding process guarantees to decode any messages contained in the packet, it also decodes extra messages, referred to as "hallucination". To control the number of hallucination messages, a checksum sequence containing l 0-bits is added into the original message. Furthermore, to control the width of the decoding tree, s bits are inserted in every f bits in the original message. We leave the detailed analysis of these parameters to the original paper [19].

3.5.2 Uncoordinate Frequency Hopping (UFH)

UFH was first proposed in [15] and was later on improved in [14, 16]. UFH works on multi-channel networks where each node changes the communication frequency randomly in each time slot without *any coordination*. In this way, nodes opportunistically communicate when they are on the same channel. The main advantage of UFH is the simplicity of the implementation. However, its major disadvantage is the high-delay/low-throughput communication. Thus, it should be used to exchange control information only.

3.6 Alibi Identification Algorithms

This section discusses several common algorithms that can be used to differentiate attackers and honest nodes based on their alibi alibi score function over time.

3.6.1 Alibi Score Function

In the alibi framework, the central detector BS keeps track of the number of alibis obtained by each node over time. Denote $ascore_r(\mathcal{T})$ to be the number of alibis obtained for node $r \in \mathcal{N}$ over the time slot set $\mathcal{T}$. At the beginning, $ascore_r(\cdot)$ is set to 0. At each time slot $t \in \mathcal{T}$, the alibi score is updated accordingly. The nodes that have lowest alibi scores consistently are likely to get accused. Because the alibi score is probabilistic by nature, we need certain statistical detection algorithms to differentiate the attackers and the honest nodes based on their alibi scores. Thus, there will be certainly

trade-offs of the detection time and the detection accuracy. In the following sections, we present two statistical detection algorithms.

3.6.2 Distance-based Outlier Detection Algorithms

Given a set of $ascore_r(\mathcal{T}), \forall r \in \mathcal{N}$ in the time slot set $\mathcal{T}$, we need to identify the set of nodes with the alibi scores that are too low compared to other nodes. Because we do not know the distribution of the alibi scores, the outlier detection algorithm has to be non-parametric. Therefore, there will certainly be the tradeoff between the false alarm probability and correct detection probability. In the alibi framework, we use a distance-based outlier detection technique as follows.

Denote μ, σ the mean and standard deviation of $ascore_r(\mathcal{T})$, respectively. A node r is determined as outlier if its distance to the "center" (i.e., μ) is larger than a pre-determined threshold ξ. We use the ***Mahalanobis squared distance*** calculated as $d(r) = (ascore_r - \mu)^2\sigma^{-1}$. Mahalanobis squared distance $d(i)$ is used rather than Euclidian distance because Mahalanobis distance normalizes the original distances into the scale-invariant distances that can be compared to the χ^2 distribution. Specifically, Mahalanobis distance has a property that the probability of $d(i) > \chi^2(\gamma)$ is γ, where $\chi^2(\gamma)$ is the upper (100γ)-th percentile of a chi-square distribution.

A node r is accused as an attacker if $d(r) < \mu$ and $d(r) \geq \xi$. The first condition ensures that we only accuse nodes that have alibi scores lower than the mean μ. The second condition specifies the threshold ξ in which r is accused based on its distance $d(r)$. Intuitively, lower value of ξ increases the detection probability (i.e., accusing r when r is an attacker), but also increases the false alarm probability (i.e., accusing r when r is an honest node). In the alibi framework, ξ is chosen based on the target false alarm probability γ. Specifically, $\xi = \chi^2(\gamma)$. For example, if the target false alarm probability γ is 0.1, ξ is set to $\chi^2(0.1) = 2.706$.

3.6.3 Sequential Hypothesis Testing

Denote $\theta_0 = ascore_r(\mathcal{T})/|\mathcal{T}|$ to be the alibi rate of an honest r. Denote $\theta_1 = ascore_j(\mathcal{T})/|\mathcal{T}|$ to be the alibi rate of an attacker j. Intuitively, if

$\theta_0 > \theta_1$, then the honest node r obtains faster alibis than an attacker and thus becomes different from the attacker. In what follows, we use sequential hypothesis testing to determine whether a node is good or bad according to its alibi rate.

Let us denote $Y_i(t)$ as a random process of obtaining alibis of node i. For node i, $Y_i(t) = 0$ if no alibi is obtained and $Y_i(t) = 1$ if an alibi is obtained. Let us denote H_0 as the hypothesis that a node is an honest node and H_1 is the hypothesis that a node is an attacker. Therefore,

$$Pr[Y_i(t) = 0|H_0] = 1 - \theta_0, Pr[Y_i(t) = 1|H_0] = \theta_0$$

$$Pr[Y_i(t) = 0|H_1] = 1 - \theta_1, Pr[Y_i(t) = 1|H_1] = \theta_1$$

The likelihood ratio in an interval $[0, T]$ is

$$\Lambda(Y_i) = \frac{Pr[Y_i|H_0]}{Pr[Y_i|H_1]} = \prod_{t=0}^{T} \frac{Pr[Y_i(t)|H_0]}{Pr[Y_i(t)|H_1]}$$

, where Y_i is the vector of events that node i observed in $[0, T]$ and $Pr[Y|H_i]$ represents the conditional probability mass function of the event stream Y_i given that H_i is true. Y_i is a random walk. The rule for the detection relies on two thresholds η_1 and η_0 ($\eta_1 > \eta_0$). If $\Lambda(Y_i) \geq \eta_1$, then the hypothesis H_0 is concluded. If $\Lambda(Y_i) \leq \eta_0$, then the hypothesis H_1 is concluded. If $\eta_0 < \Lambda(Y_i) < \eta_1$, more observations are needed to make the conclusion.

Intuitively, η_0 and η_1 directly affect the false alarm probability and detection probability. Let us consider false alarm probability P_F and detection probability P_D. Let us denote the user-chosen values of the false alarm and detection probability are α and β, respectively. That means, $P_F \leq \alpha$ and $P_D \geq \beta$. We need to find out the relationship between α, β and η_0, η_1.

In [20], Wald proves that if we set $\eta_1 \leftarrow \frac{\beta}{\alpha}$ and $\eta_0 \leftarrow \frac{1-\beta}{1-\alpha}$, then the **actual** false alarm P_F^* and detection probability P_D^* have following relationships: 1) $P_F^* < \frac{\alpha}{\beta}$; 2) $1 - P_D^* < \frac{1-\beta}{1-\alpha}$ and 3) $1 - P_D^* + P_F^* \leq 1 - \beta + \alpha$. He further shows that the expected number of samples T to conclude the hypothesis H_1 is

$$E[T|H_1] = \frac{\beta \ln \frac{\beta}{\alpha} + (1-\beta) \ln \frac{1-\beta}{1-\alpha}}{\theta_1 \ln \frac{\theta_1}{\theta_0} + (1-\theta_1) \ln \frac{1-\theta_1}{1-\theta_0}}.$$

CHAPTER 4

ALIBI DESIGN FOR MULTI-CHANNEL WLANS

4.1 Overview

In this chapter, we present a design of an alibi framework for multi-channel WLANs. The network has the set of nodes $\mathcal{N} = 1, \ldots, |\mathcal{N}|$ and a trusted base station. Nodes communicate with the base station as illustrated in Figure 4.1.

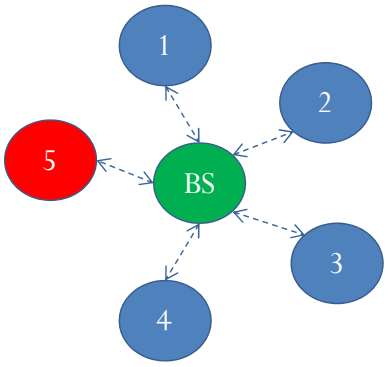


Figure 4.1: Network Model

The network uses Time-Division Multiple Access (TDMA). The base station delivers the TDMA schedule to the nodes over a *control channel*. We assume the base station can send/receive in all channels simultaneously. However, we assume an abstract assignment as follows. There will be n *slots* of size s in a *round*. Each slot is uniquely pre-assigned to a node. Honest nodes only transmit in their assigned time slots. In the normal situation, the TDMA scheduling has no collisions. Figure 4.2 gives an example of a TDMA schedule on 3 channels and 6 nodes. Slots 1 and 7 are the slots for the base station. The rest of slots is for each node i to send to the base station.

The basic idea is to exploit the limited capabilities of the jammer j: he cannot jam two channels simultaneously. This limitation implies that if he

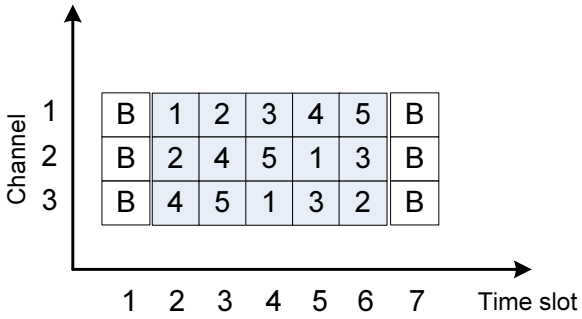


Figure 4.2: Example of a TDMA scheduling for 6 nodes on 3 channels. The number in the box denotes the sender at that time slot. B means the base station.

jams on one channel in a time slot, he cannot jam any other channel in the same time slot. This opens a chance for good nodes to *prove their honesty*. Nodes can transmit on other channels to prove that they were transmitting on another channel while one channel was jammed in a time slot. Such sending proofs are referred to as *sending-based alibis (s-alibis)*. Similarly, if a node receives an uncorrupted packet content in one channel while there is another jamming activity in another channel, it will get a *receiving-based alibi*. The important property of alibis is that *only good nodes can prove their honesty while the jammer can never do that*. Eventually, all good nodes are proved to be honest and the jammer j is identified.

In the design of the alibi framework, sending-based alibis are used to identify insider jammers in the uplink traffic. Receiving-based alibis are used to identify insider jammers in the downlink traffic. The design of this framework, despite of its simplicity, demonstrates the applications of sending-based and receiving-based alibis.

4.2 S-alibis for Uplink Traffic

For the *up-link traffic* (i.e., the traffic from nodes to the base station), nodes have chances to become the defendants and the base station is the only witness. Figure 4.3 shows an example of s-alibis for a network of 5 nodes and 3 channels. At time slot 1, the base station detects a jammed packet at channel 1 while receiving uncorrupted packets from node 2 and 4 on channel 2 and 3, respectively. Therefore, node 2 and 4 will get a s-alibi. Similar,

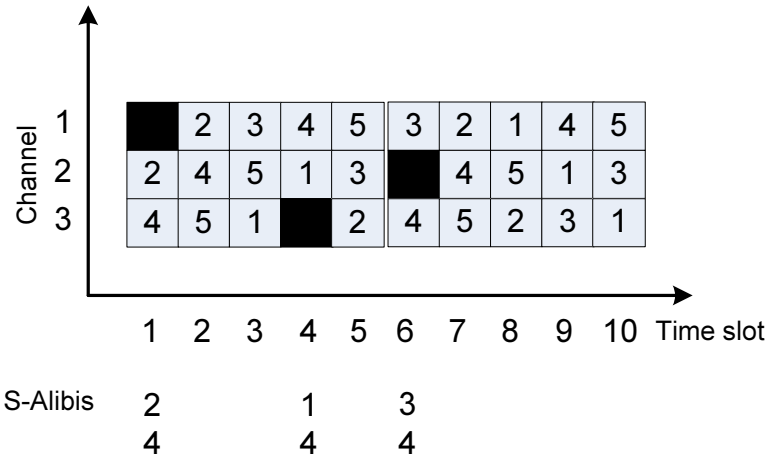


Figure 4.3: Example of the S-alibis for up-link traffic. At time slot 1, 4 and 6, there are jammed packets. Thus, nodes (2,4), (1,4) and (3,4) get s-alibis. 5 is accused as the insider jammer due to no s-alibis obtained.

for the time slots 4 and 6, nodes 1,4 and 3,4 will get a s-alibi. In this way, whenever there is jammed packet, there will be at least some nodes getting s-alibis. Eventually, the nodes which have the lowest number of s-alibis are accused as the insider jammers.

4.3 R-alibis for Downlink Traffic

The situation is different for the *down-link traffic* (i.e., the broadcast message from the base station to the nodes). Because the base station broadcasts at all channels, nodes listening on the un-jammed channels will receive a correct message while nodes listening on the jammed channels will receive a corrupted packet. Each node keeps received packets as *proofs*. In a time slot, if there are some nodes U receiving corrupted packets and some other nodes V receiving uncorrupted broadcast message, the nodes V will receive R-alibis.

In this way, nodes are required to send to the base station the content of their last received broadcast message (i.e., proofs). The base station will decide to give a R-alibi to a node based on the content of that node's last received broadcast messages. Specifically, if the received content by a node s_1 matches with the content of the broadcast message and there is a corrupted content reported by another node s_2, s_1 will get a R-alibi.

Figure 4.4 gives an example of the R-alibis for down-link traffic. In the first downlink slot, channel 1 is jammed. Nodes 1 and 2 receive an uncorrupted broadcast message from the base station and may get an R-alibi. Similarly, the second and third downlink slot are jammed. Nodes 2, 3 and 3, 4 may get an R-alibi. The R-alibis are actually obtained when nodes send the proofs back to the base station during proof-collection period.

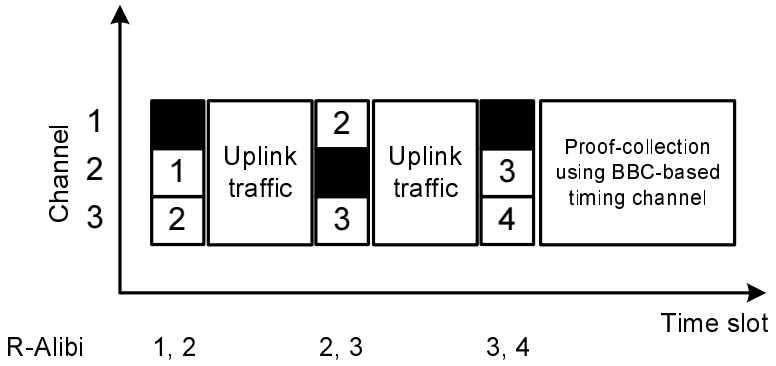


Figure 4.4: Example of the R-alibis for down-link traffic.

4.4 Alibi Protocols

For the up-link traffic, the proofs are already collected by the base station. Therefore, the network does not need to have special protocol to collect proofs in case of the up-link traffic. For the down-link traffic, nodes have to send the proofs collected from the last broadcast message to the base station. Because the normal up-link traffic may be jammed, the proofs have to be transmitted using a low-throughput jamming-resistant communication channel. Specifically, the proof transmission period is triggered when the base station detects a jamming attack on the down-link traffic. The base station uses a BBC-based jamming-resistant channel to broadcast a *proof-collection* message to all nodes. Each node, upon receiving the *proof-collection* message, will start to transmit proofs that they have collected to the base station using the BBC-based jamming resistant channel.

4.4.1 Jamming Detection in the Downlink Traffic

To detect a jamming attack on the down-link traffic, the base station counts the number of *un-acknowledged* messages (referred to as the *jamming counter*). When the the jamming counter exceeds a jamming threshold, the base station declares a jamming attack. The jamming threshold is a tunable parameter.

A down-link message is marked as un-acknowledged as follows.

- If it is a *uni-cast* message to a node, the jamming counter is simply set (by the base station) to the number of unicast messages that do not get acknowledged by the destination nodes.

- If it is a *multi-cast/broadcast* message, the nodes receiving it, regardless of the correctness of the received content, will piggy-back a *hash* of the content in their next data packets. The jamming counter is set (by the base station) to sum of the number of nodes reporting the un-matched hashes of the broadcast content and the number of nodes *not* reporting the hash of the broadcast content. The former concerns about the nodes that receive corrupted broadcast messages and *can* send back the hash of the received content. The latter concerns about the nodes that receive corrupted broadcast and *cannot* send back the hash of the received content due to the up-link jamming attacks. Figure 4.5 shows an example of the jamming detection in the down-link traffic. In the example, the base station broadcasts a message m. Node 1 receives m correctly. Nodes 2, 3 and 4 receive m_2, m_3 and m_4, which are corrupted versions of m, respectively. All nodes attach a hash of the content of their received broadcast messages $H(m_2), H(m_3), H(m_4)$ in the next data packets they send to the base station. Upon receiving the messages from nodes 2 and 4 which contains $H(m_2), H(m_4)$, the base station increases the jamming counter by 2 because the messages from nodes 2 and 4 are received correctly and 2) the hashes $H(m_2), H(m_4)$ do not match with $H(m)$. Upon receiving the message from node 1 containing $H(m_1)$, the base station does not increase the jamming counter because the message from node 1 is received correctly and the embedded hash content matches with $H(m)$. The hash reported by node 3 is not received correctly by the base station and thus does not count. Note that if nodes 2, 3 and 4 send some messages to the base station

and embed $H(m_2), H(m_3), H(m_4)$, the base station will not count them again (i.e., nodes 2 and 4). For the node 3, it will be eventually counted until the next broadcast message m' because node 3 does not report correctly what it received for the last broadcast message m.

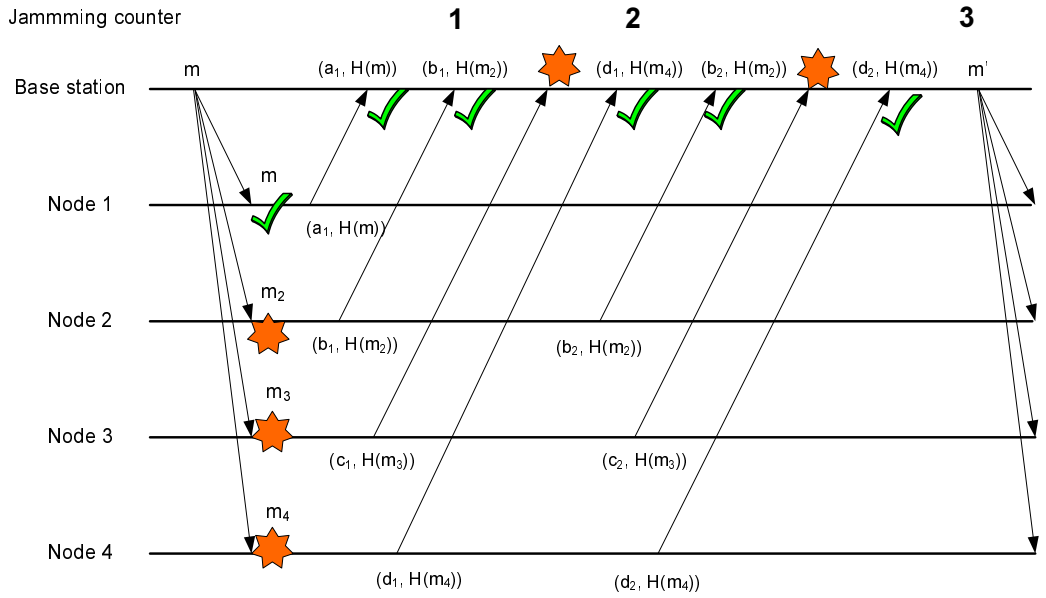


Figure 4.5: Example of the jamming detection in the down-link traffic. Checking symbols represent successful transmissions. Star symbols represent corrupted transmission.

The hash function H used to hash the received content of the broadcast message can be any hash function such as SHA1 or MD5. It is just used to verify the integrity of the received message. Note that there is another way to implement the jamming detection protocol. The base station can simply attach a message integrity code (MIC) in the message m. The nodes can verify whether they received correct broadcast messages or not. Then, they can attach one bit result in the messages to the base station indicating whether they received correct broadcast messages or not.

4.4.2 BBC-based Control Channel

The base station broadcasts the *proof-collection* message to nodes using the BBC-based control channel (see Chapter 3). It only broadcasts this message when it detects a jamming situation in the down-link traffic. It will also

broadcast this message in *all* channels. Therefore, any node, regardless of which channels they are in, is able to receive the *proof-collection* message. This ensures no nodes miss the proof-collection period.

Note that the transmission of the proof-collection message may be long due to the low-throughput nature of the timing channel. Thus, it may last over the up-link period. That means, nodes may not be able to have successful transmission to the base station. However, this is just a temporary period where the base station preempts the channel to send the important control message.

When nodes receive the proof-collection message, they will also use BBC-based control channel to send proofs to the base station. Specifically, each node will randomly select a channel in $\mathcal{C}$ on which it uses the BBC-based timing communication to transmit proofs. Because the base station can listen to all channels, it will be able to receive all proofs from nodes.

It is also important to emphasize that when nodes receive the proof-collection message, they will first estimate the start of the broadcast period and only report the proofs *before* that period. This is to avoid reporting the false corruption caused by the base station broadcasting the proof-collection message.

4.5 Detectability Analysis for Uplink Traffic

In the following sub-section, we are going to analyze the detectability of the insider jammers for the uplink traffic. Specifically, we want to analyze the *necessary* conditions that allow the alibi system to identify the attackers. That means, if the necessary condition is not met, the system cannot make any guarantee in identifying any attackers. Furthermore, if the necessary condition is met, the system can promise to identify the attackers, given the following assumptions:

- There are no channel losses. In other words, any corrupted packets are the results of the jamming actions.
- All nodes, including compromised nodes, have non-zero sending rate. Furthermore, nodes, including compromised nodes, are assumed to always have packets to send.

- Whenever a jammer decides to jam at a time slot, it will jam on a uniformly selected channel in $\mathcal{C}$.
- In any time slot, two events, (a) "a particular jammer jams", and (b) "a particular sender sends", are independent.
- Nodes, including compromised nodes, will fully utilize the slots allocated for their transmission. That means, there are no "idle" allocated transmission slots.

Theorem 1 (Identifying non-colluding attackers in the up-link traffic) *In the network $\mathcal{N}$ with the set of non-colluding jammers $\mathcal{J} = j_1, \ldots, j_{|\mathcal{J}|}$ with the allocated sending rates $p_{j_1}^{send}, \ldots, p_{j_{|\mathcal{J}|}}^{send}$ and the jamming rates $p_{j_1}^{jam}, \ldots, p_{j_{|\mathcal{J}|}}^{jam}$, the alibi framework can identify any jammer j $(j \in \mathcal{J}, p_j^{jam} > 0)$ if*

$$p_j^{jam}(\frac{b - a/|\mathcal{C}|}{a - b} - \frac{wb}{1 - b}) > w - 1$$

where $w = \frac{p_j^{send}}{p_{smin}^{send}}$, p_{smin}^{send} is the minimum sending rate of all nodes in $\mathcal{N}$,

$$a = \prod_{i \in \mathcal{J} \setminus j} (1 - \frac{p_i^{jam}}{|\mathcal{C}|})$$

and

$$b = \prod_{i \in \mathcal{J} \setminus j} (1 - p_i^{jam})$$

.

Proof:

Let us consider an honest node $s \in \mathcal{N} \setminus \mathcal{J}$ with sending rate p_s^{send} and a jammer $j \in \mathcal{J}$ with the sending rate p_j^{send}. Let us denote p_j^{jam} the jamming rate of jammer j ($p_j^{send} + p_j^{jam} \leq 1$) because j cannot simultaneously jam and send at the same time. In other words, the sending events and the jamming events of j are mutual exclusive.

For an honest s, in any time slot, with probability p_s^{send}, s will send a packet. When s sends a packet, it will obtain s-alibi if at least one attacker in $\mathcal{J}$ jams and its sending packet is not jammed. Therefore, the probability that s obtains s-alibi in a time slot is

$p_s^{send} \times Pr[\text{no attackers jam on a particular channel} \mid \text{at least one attacker jams}] =$

$$p_s^{send} \times Pr[A(\mathcal{J})|\bar{B}(\mathcal{J})]$$

where $A(\mathcal{J})$ denotes the event of "no attackers in $\mathcal{J}$ decides to jam on a particular channel" , $B(\mathcal{J})$ denotes the event of "no attacker in $\mathcal{J}$ decides to jam" and $\bar{B}(\mathcal{J})$ denotes the event of "at least one attacker in $\mathcal{J}$ decides to jam". It is easy to see that

$$Pr[A(\mathcal{J})] = \prod_{i \in \mathcal{J}} (1 - \frac{p_i^{jam}}{|\mathcal{C}|})$$

and

$$Pr[B(\mathcal{J})] = 1 - Pr[\bar{B}(\mathcal{J})] = \prod_{i \in \mathcal{J}} (1 - p_i^{jam})$$

.

By using the conditional probability expansion, we have

$$Pr[A(\mathcal{J})|\bar{B}(\mathcal{J})] = \frac{Pr[A(\mathcal{J}) \cap \bar{B}(\mathcal{J})]}{Pr[\bar{B}(\mathcal{J})]} = \frac{Pr[A(\mathcal{J}) - B(\mathcal{J})]}{1 - Pr[B(\mathcal{J})]}$$

. Since $B(\mathcal{J})$ is a subset of $A(\mathcal{J})$, we have $Pr[A(\mathcal{J}) - B(\mathcal{J})] = Pr[A(\mathcal{J})] - Pr[B(\mathcal{J})]$.

Therefore, in any time slot, s will get a s-alibi with probability

$$p_s^{send} \frac{Pr[A(\mathcal{J})] - Pr[B(\mathcal{J})]}{1 - Pr[B(\mathcal{J})]} \tag{4.1}$$

Similarly, in any time slot, with probability p_j^{send}, j will send a packet and obtain s-alibi given that at least one attacker in $\mathcal{J} \backslash j$ jams and j's packet is transmitted successfully. By doing a similar analysis, we can derive that the probability that j obtains s-alibi in a time slot is

$$p_j^{send} \frac{Pr[A(\mathcal{J} \backslash j)] - Pr[B(\mathcal{J} \backslash j)]}{1 - Pr[B(\mathcal{J} \backslash j)]} \tag{4.2}$$

.

For s to obtain s-alibis faster than j, we need

$$p_s^{send} \frac{Pr[A(\mathcal{J})] - Pr[B(\mathcal{J})]}{1 - Pr[B(\mathcal{J})]} > p_j^{send} \frac{Pr[A(\mathcal{J} \backslash j)] - Pr[B(\mathcal{J} \backslash j)]}{1 - Pr[B(\mathcal{J} \backslash j)]}$$

. To identify j, we need to ensure the above condition for *all* honest nodes. That means, we need

$$p_{smin}^{send}\frac{Pr[A(\mathcal{J})]-Pr[B(\mathcal{J})]}{1-Pr[B(\mathcal{J})]} > p_j^{send}\frac{Pr[A(\mathcal{J}\backslash j)]-Pr[B(\mathcal{J}\backslash j)]}{1-Pr[B(\mathcal{J}\backslash j)]}$$

where p_{smin}^{send} is the minimum sending rate of all honest nodes.

Denote $a = Pr[A(\mathcal{J}\backslash j)]$, $b = Pr[B(\mathcal{J}\backslash j)]$, $p = p_j^{jam}$, $w = \frac{p_j^{send}}{p_s^{send}}$ and $c = |\mathcal{C}|$. We have $Pr[A(\mathcal{J})] = (1-p/c)a$ and $Pr[B(\mathcal{J}\backslash j)] = (1-p)b$. Thus, the above condition becomes

$$\frac{(1-p/c)a-(1-p)b}{1-(1-p)b} > w\frac{a-b}{1-b}$$

. By expanding the above inequality, we obtain

$$a-\frac{p}{c}a-b+pb-ba+\frac{p}{c}ba+b^2-pb^2 > wa-wab+wpab-wb+wb^2-wpb^2$$

. After some manipulations, we get

$$p((1-b)(b-\frac{a}{c})-wb(a-b)) > (a-b)(1-b)(w-1)$$

. Divide both sides by the term $(a-b)(1-b)$, we have

$$p(\frac{b-a/c}{a-b}-\frac{wb}{1-b}) > w-1$$

.

$\diamond$

The difference for the case of colluding attackers is that they can coordinate with each other to avoid overlapping jamming in any channel. Specifically, they coordinate as follows. In any time slot, if an attacker decides to jam, it will jam on the channel that 1) does not have any other attackers jamming on and 2) does not have any other attackers sending on.

Theorem 2 (Identifying colluding attackers in the up-link traffic) *In the network $\mathcal{N}$ with the set of non-colluding jammers $\mathcal{J} = j_1, \ldots, j_{|\mathcal{J}|}$ with the sending rates $p_{j_1}^{send}, \ldots, p_{j_{|\mathcal{J}|}}^{send}$ and the jamming rates $p_{j_1}^{jam}, \ldots, p_{j_{|\mathcal{J}|}}^{jam}$, the*

alibi framework can identify any jammer j ($j \in \mathcal{J}$) if

$$p_{smin}^{send} \frac{a-b}{1-b} > (1 - p_j^{jam}) - b$$

where p_{smin}^{send} is the minimum sending rate of all honest nodes in $\mathcal{N}$,

$$a = \frac{|\mathcal{C}| - |\mathcal{J}|}{|\mathcal{C}| - \sum_{i \in \mathcal{J}} (1 - p_i^{jam})}$$

and

$$b = \prod_{i \in \mathcal{J}} (1 - p_i^{jam})$$

even under the best jamming rates of the attackers where $p_{j_1}^{jam} = 1 - p_{j_1}^{send}, \ldots, p_{j_{|\mathcal{J}|}}^{jam} = 1 - p_{j_{|\mathcal{J}|}}^{send}$.

Proof:

From the coordination strategy above, it is intuitive to see that in any time slot, if one attacker decides to jam, it is best for all other non-sending attackers also decide to jam. This is because if a non-sending attackers does not jam when at least another attacker jams, it gives up a chance to block a channel that has an honest node sending on.

Denote p_T the desired jamming probability of the attackers. That means, in any time slot, with probability p_T, all non-sending attackers decide to jam and with probability $1 - p_T$, all non-sending attackers decide not to jam. Because we are only concerned about the detectability of the attackers, we will only consider the time slots that all non-sending attackers decide to jam. That means, in the following analysis, we assume that in each time slot all non-sending attackers will decide to jam coordinately. In other words, for any attacker $j \in \mathcal{J}$, $p_j^{jam} = 1 - p_j^{send}$.

Let us now consider an honest node $s \in \mathcal{N} \backslash \mathcal{J}$ with the sending rate p_s^{send} and a jammer $j \in \mathcal{J}$ with the sending rate p_j^{send} and the jamming rate $p_j^{jam} = 1 - p_j^{send}$.

At a time slot, if s decides to send, the probability that s sends a successful message to the base station and thus gets a s-alibi is

$p_s^{send} \times Pr[\text{no attackers jam on a particular channel} \mid \text{at least one attacker jams}] =$

$$p_s^{send} \times Pr[A(\mathcal{J}) | \bar{B}(\mathcal{J})] = p_s^{send} \times Pr[A(\mathcal{J}) \cap \bar{B}(\mathcal{J})] / P[\bar{B}(\mathcal{J})]$$

where $A(\mathcal{J})$ denotes the event of "no attackers in $\mathcal{J}$ jam on a particular channel" , $B(\mathcal{J})$ denotes the event of "no attacker in $\mathcal{J}$ decides to jam" and $\bar{B}(\mathcal{J})$ denotes the event of "at least one attacker in $\mathcal{J}$ decides to jam". It is easy to see that

$$Pr[B(\mathcal{J})] = 1 - Pr[\bar{B}(\mathcal{J})] = \prod_{i \in \mathcal{J}} (1 - p_i^{jam})$$

. We now need to calculate $Pr[A(\mathcal{J})]$ under the colluding strategy of the attackers in $\mathcal{J}$.

Let us consider a specific time slot where s sends. The expected number of attackers sending in this time slot is $\sum_{i \in \mathcal{J}} p_i^{send}$, which is also the expected number of channels that the attackers use to send as scheduled. Thus, the remaining $(|\mathcal{C}| - \sum_{i \in \mathcal{J}} p_i^{send})$ channels is the channel set that the attackers may jam on and one of the channel is the channel that s is sending on. Also, $(|\mathcal{J}| - \sum_{i \in \mathcal{J}} p_i^{send})$ is the expected number of attackers deciding to jam. Furthermore, those attackers only jam (uniformly) in the remaining $(|\mathcal{C}| - \sum_{i \in \mathcal{J}} p_i^{send})$ channels. Thus, the probability that the channel that s sends on does not get jammed is

$$Pr[A(\mathcal{J})] = \frac{(|\mathcal{C}| - \sum_{i \in \mathcal{J}} p_i^{send}) - (|\mathcal{J}| - \sum_{i \in \mathcal{J}} p_i^{send})}{|\mathcal{C}| - \sum_{i \in \mathcal{J}} p_i^{send}} = \frac{|\mathcal{C}| - |\mathcal{J}|}{|\mathcal{C}| - \sum_{i \in \mathcal{J}} (1 - p_i^{jam})}$$

.

We have

$$Pr[A(\mathcal{J}) \cap \bar{B}(\mathcal{J})] = Pr[A(\mathcal{J})] - Pr[A(\mathcal{J}) \cap B(\mathcal{J})]$$

. Since

$$Pr[A(\mathcal{J}) \cap B(\mathcal{J})] = Pr[A(\mathcal{J})|B(\mathcal{J})] \times Pr[B(\mathcal{J})]$$

and $Pr[A(\mathcal{J})|B(\mathcal{J})] = 1$ (because the probability of no attackers jamming on a particular channel given that no attackers deciding to jam is equal to 1), we have

$$Pr[A(\mathcal{J}) \cap \bar{B}(\mathcal{J})] = Pr[A(\mathcal{J})] - Pr[B(\mathcal{J})]$$

. Thus, the probability s gets an s-alibi in any time slot is

$$p_s^{send} \times \frac{Pr[A(\mathcal{J})] - Pr[B(\mathcal{J})]}{P[\bar{B}(\mathcal{J})]} = p_s^{send} \frac{Pr[A(\mathcal{J})] - Pr[B(\mathcal{J})]}{1 - P[B(\mathcal{J})]}$$

Similarly, for the attacker j, it obtains a s-alibi when it sends and at least another attacker jams. That probability is

$$p_j^{send} Pr[\mathcal{B}(\mathcal{J}\backslash j)] = p_j^{send} \times (1 - \prod_{i \in \mathcal{J}\backslash j} (1 - p_i^{jam})) = (1 - p_j^{jam}) \times (1 - \prod_{i \in \mathcal{J}\backslash j} (1 - p_i^{jam}))$$

.

Let us denote $a = Pr[\mathcal{A}(\mathcal{J})]$ and $b = Pr[\mathcal{B}(\mathcal{J})]$. Thus, we have

$$p_s^{send} \frac{Pr[A(\mathcal{J})] - Pr[B(\mathcal{J})]}{1 - P[B(\mathcal{J})]} = p_s^{send} \frac{a - b}{1 - b}.$$

and

$$(1 - p_j^{jam}) \times (1 - \prod_{i \in \mathcal{J}\backslash j} (1 - p_i^{jam})) = (1 - p_j^{jam}) - \prod_{i \in \mathcal{J}} (1 - p_i^{jam}) = (1 - p_j^{jam}) - b.$$

Thus, in order to have node s obtains S-alibis faster than j, we need

$$p_s^{send} \frac{a - b}{1 - b} > (1 - p_j^{jam}) - b$$

.

$\diamond$

4.6 Detectability Analysis for Downlink Traffic

The case of the down-link traffic depends on whether nodes can successfully send proofs to the base station. Once proofs are correctly received by the base station, the analysis is similar to the case of the up-link traffic. However, the detection time will be longer due to two reasons. First, the base station may have less number of transmissions than that of all the nodes. Second, the proofs have to be sent back to the base station using a low throughput channel.

4.7 Dealing with Slander Attacks on Uplink Traffic

The alibi framework on a simple TDMA scheduling has one drawback: it cannot defend against the slander attack on the up-link traffic. In the slander

Parameter	Values
TDMA slot size	500ms
Number of channels	10
Number of nodes	$n = [10 - 40]$
Number of attackers	$[1 - 9]$
Jamming rate	$[0.1 - 1.0]$
γ	0.001
Simulation time	1000 seconds
BBC maximum number of concurrent transmissions	50
BBC message length in time	4s
Number of BBC messages per proof-exchange period	10

Table 4.1: Simulation parameters

attack, the attacker chooses a set of victim nodes and does not jam whenever a victim node becomes a sender. This attack will prevent any victim nodes to get s-alibis because there is no jamming activities whenever the victim nodes become the sender.

To deal with the slander attacks, it is required that the attackers do not know *when* a node becomes a sender. Specifically, the base station has to ensure the following two properties of the TDMA schedule for nodes.

1. *Randomness*: The schedule has to be random for each period.

2. *Confidentiality*: The schedule of a node is known by that node and the base station only.

Both properties ensure that the attacker cannot guess which slots belong to a node and thus cannot perform the slander attacks. The first property can be done by having the base station randomly shuffle the TDMA schedule for each period. The second property can be ensured by encrypting the schedule using they key of the destine node.

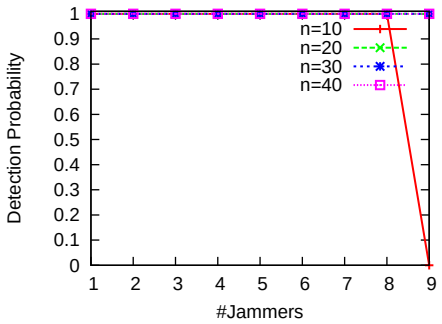
4.8 Evaluation

We evaluate the proposed protocols in TOSSIM. The TDMA MAC is implemented by modifying the existing MAC of MICAz. The simulation parameters are listed in Table 4.1. Each scenario is repeated 5 times to get the confident statistics.

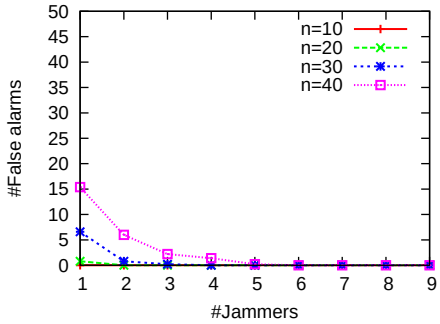
4.8.1 Non-colluding Attackers

Figures 4.6 and 4.7 show the performance of the alibi framework in identifying non-colluding attackers. Specifically, Figure 4.6 shows the performance of the alibi framework for the case where all attackers with probability 1.0. Particularly, Figure 4.6(a) shows a high accuracy detection probability while achieving low number of false alarms (Figure 4.6(b)). Figures 4.6(c) and 4.6(d) further show the detection time of the framework and the network performance in terms of packet error rate, respectively.

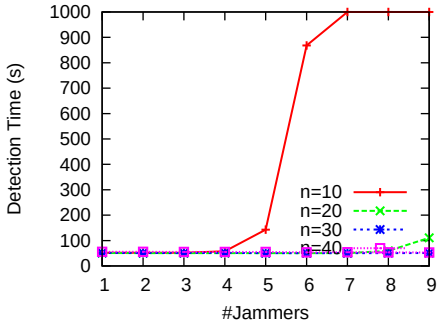
Figure 4.7 shows the performance of the alibi framework for the case of the network size $n = 40$. As shown in the figures, higher jamming rates will lead to better detection accuracy because more alibis can be granted to the honest nodes.



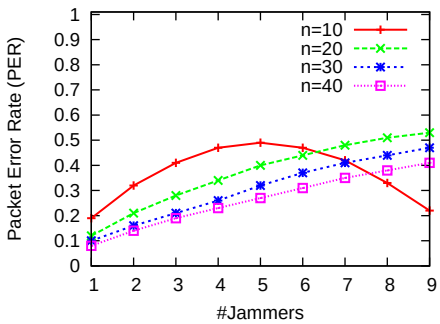
(a) Detection probability (non-colluding), pj=1



(b) False alarms (non-colluding), pj=1

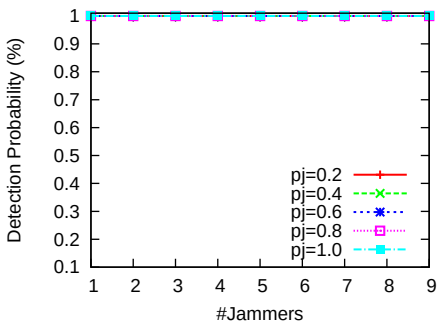


(c) Detection Time (non-colluding), pj=1

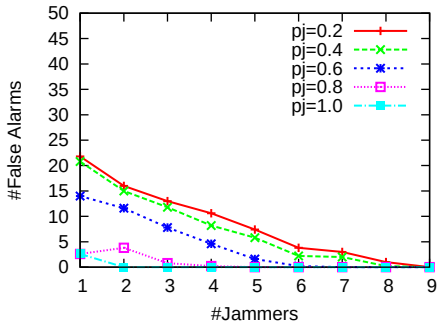


(d) Packet error rate (non-colluding), pj=1

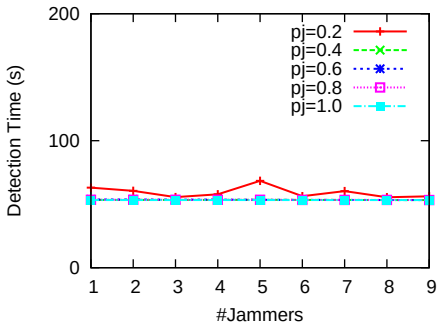
Figure 4.6: Performance for the case of non-colluding attackers, pj=1



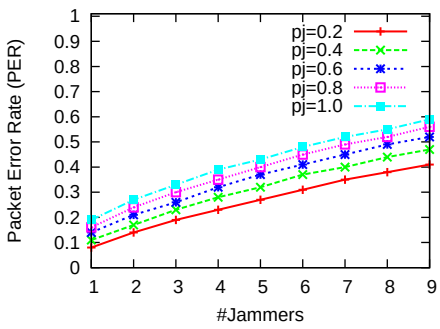
(a) Detection probability (non-colluding), n=40



(b) False alarms (non-colluding), n=40



(c) Detection Time (non-colluding), n=40



(d) Packet error rate (non-colluding), n=40)

Figure 4.7: Performance for the case of non-colluding attackers, n=40

4.8.2 Colluding Attackers

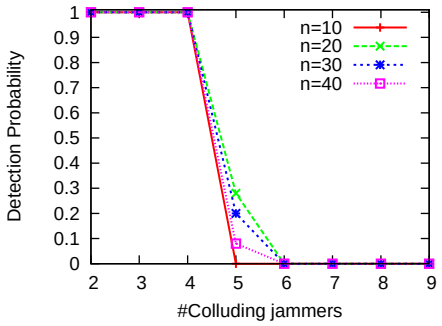
We also evaluate the performance of the alibi framework for the case of colluding attackers. In the evaluation, the colluding attackers are coordinated. They have a smart jamming strategy as follows.

- The jammers aim at a specific total jamming target and distribute the jamming effort evenly. They never have overlapping jamming (i.e., no two jammers will jam the same channel). For example, if the total jamming target is 1.0 and there are 5 jammers, each jammer will jam 0.2 on average.

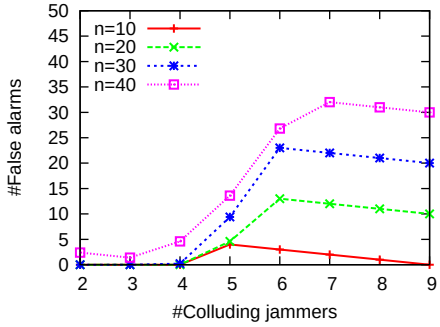
- In each time slot, the jammers decide to jam or not according the total jamming target. If they decide to jam, each of the jammer selects a random channel to jam. If the selected channel happens to be the channel that another channel is supposed to send the uplink traffic (not the channel that another jammer will jam), the selecting jammer will act on behalf o that jammer and sends a legitimate packet to get an S-alibi. This strategy will guarantee that if a jammer does not have a useful jamming, i.e., jamming on the uplink channel of an honest node, it will send a legitimate packet and get an alibi instead.

Figure 4.8 shows the performance of the alibi framework for the case of colluding attackers when the total jamming target is 1. Specifically, as shown in Figures 4.8(a) and 4.8(c), the system cannot cope with more than 5 attackers. The reason is that when the number of attackers is more than half of the number of channels, they can always manage to have more S-alibis than other nodes (see Theorem 2). Figures 4.8(c) and 4.8(d) show the detection time and the packet error rate, respectively.

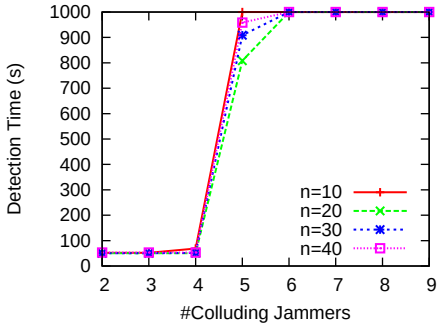
Figure 4.9(a) shows the the performance of the alibi framework for the case of colluding attackers when the network size is 40 and the total jamming target is varied. As shown in Figure 4.9(a) and 4.9(b), when the total jamming target is less, the system can cope with more attackers. Figures 4.9(c) and 4.9(d) show the detection time and the packet error rate, respectively. The overall results show that it is much harder to identify colluding attackers than to identify non-colluding attackers due to the knowledge of the alibi protocols and the coordination among the attackers.



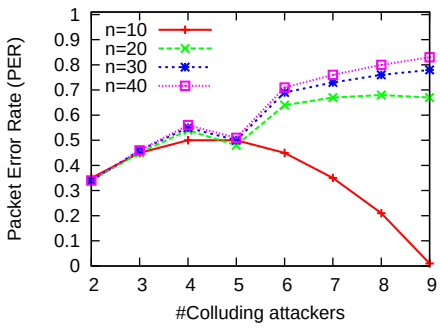
(a) Detection probability (colluding), pj=1



(b) False alarms (colluding), pj=1

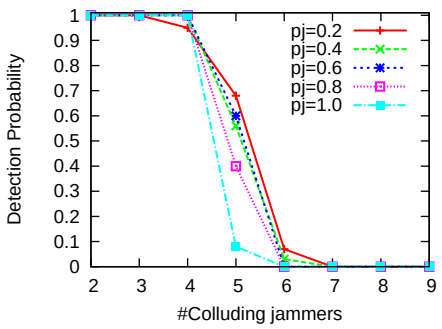


(c) Detection Time (colluding), pj=1

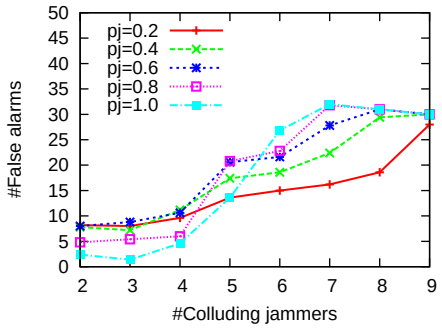


(d) Packet error rate (colluding), pj=1

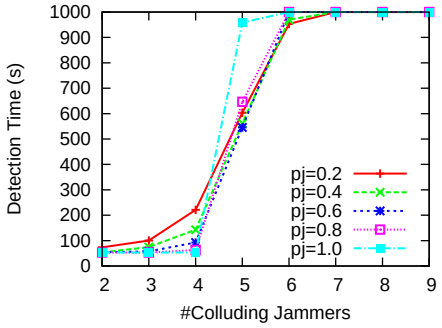
Figure 4.8: Performance for the case of colluding attackers, pj=1



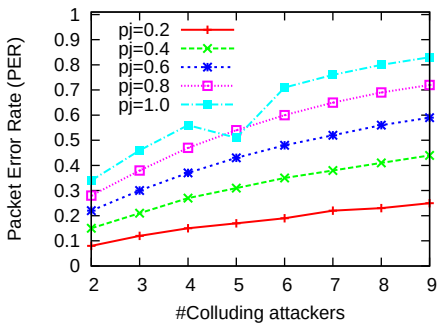
(a) Detection probability (colluding), n=40



(b) False alarms (colluding), n=40



(c) Detection Time (colluding), n=40



(d) Packet error rate (colluding), n=40)

Figure 4.9: Performance for the case of colluding attackers, n=40

Notation	Values
$\mathcal{N}$	the set of nodes in the network
$\mathcal{J}$	the set of jammers ($\mathcal{J} \subset \mathcal{N}$)
$\mathcal{C}$	the set of channels
p_j^{jam}	The jamming rate of attacker j
p_j^{send}	The sending rate of a node j
p_{smin}^{send}	the minimum sending rate of all honest nodes

Table 4.2: Table of notations used in this chapter

4.9 Discussions

The proposed design of the alibi framework has the following advantages.

- It is simple but yet can identify the attackers jamming in most of the up-link traffic and down-link traffic scenarios.
- It can deal with colluding attackers and slander attacks.

The framework also has the following disadvantages.

- It requires the base station to be able to send/receive simultaneously in all channels.
- It can only work for TDMA scheduling.
- It cannot work when the number of colluding attackers is more than half of the number of nodes or more than half of the number of channels.

CHAPTER 5

R-ALIBI DESIGN FOR MULTI-CHANNEL SINGLE-HOP AD-HOC NETWORKS

5.1 System Model

5.1.1 Network Model

We consider a multi-channel single-hop ad hoc wireless network as shown in Figure 5.1. Some nodes in the network are insider jammers. We assume that these jammers can affect at least several nodes (if not all) of the considered wireless network. We assume there is one trusted node G in the network. The trusted node can be elected by any leader election algorithm such as the one proposed in [21].

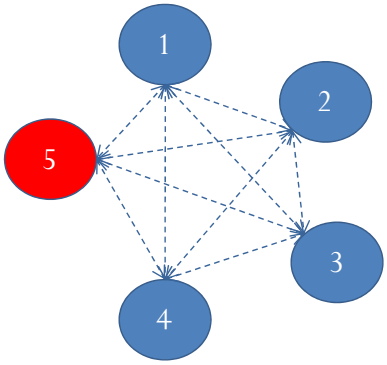


Figure 5.1: Network Model

Denote $\mathcal{N} = 1, \ldots, |\mathcal{N}|$ the set of nodes in the network and n the number of nodes ($n = |\mathcal{N}|$). Nodes in the network use a multi-channel MAC protocol such as Slotted Seeded Channel Hopping (SSCH) [22] or McMAC [23]. The main reasons for the suggestion of these multi-channel MAC protocols are:

1. These MAC protocols improve the capacity of the wireless networks. They generally do not require any special hardware other than a commodity wireless cards such as 802.11.

2. They are multiple-rendezvous protocols in which multiple device pairs can make agreements simultaneously on distinct channels [24]. This eliminates the problem of single control channel bottleneck - a sweet spot for the jammer to target on. This is very important because a jammed control channel may drastically reduce the effective throughput close to zero [12][13].

In the design of this framework, we use SSCH [22]. In SSCH, each node i has a set of n_i^{ssch} randomly generated channel seeds. Each channel seed is a pair of *(x, a)*, where x ($x \in \mathcal{C}$) is the initial channel and a ($a \in \mathcal{C}$) is the seed of the schedule. Each channel seed is used to calculate the new channel from the old channel in each time slot. Specifically, the new channel x_{new} is calculated from the old channel x_{old} as

$$x_{new} = (x_{old} + a) \mod |\mathcal{C}|.$$

, where $|\mathcal{C}|$ is the number of channels.

Channel seeds are used in round-robin manner. Specifically, in seed slot s ($s = 0, \ldots, n_i^{ssch} - 1$), node i will hop to the channel calculated from the channel seed s-th. Figure 5.2 illustrates SSCH schedules for two nodes A and B. Node A has two channel seeds $(1, 2)$ and $(2, 1)$. Node B also has two channel seeds $(1, 2)$ and $(0, 1)$. A and B use the two channel seeds alternatively in each time slot. Initially, in the first time slot, A uses the first channel seed $(1, 2)$ and thus goes to channel 1. Similarly, B goes to channel 1 in the first time slot. In the second time slot, A and B will use the second channel seed. That means, A, B will go to channel $2, 0$, respectively. In the third time slot, A will use the first channel seed $(1, 2)$. Because the old channel corresponding to this channel seed that A used is 1, the new channel that A uses is $(1 + 2) \mod 3 = 0$. Similarly, in the third time slot, B's new channel is $(1 + 2) \mod 3 = 0$. For the time slot $4, 5, 6$, the new channels for A and B are calculated in a similar manner. The time slot 7 (i.e., the slot with shaded background) is the parity slot in which the channels that A and B use are set to the seed being used, instead of the calculated channel like previous time slots. The reason for the parity time slots is to ensure that any pair of nodes will meet occasionally, even when their channel seeds have different initial channel (i.e., x_i) and the same seed (i.e., a_i).

If the number of channels is a prime number, SSCH schedules have a

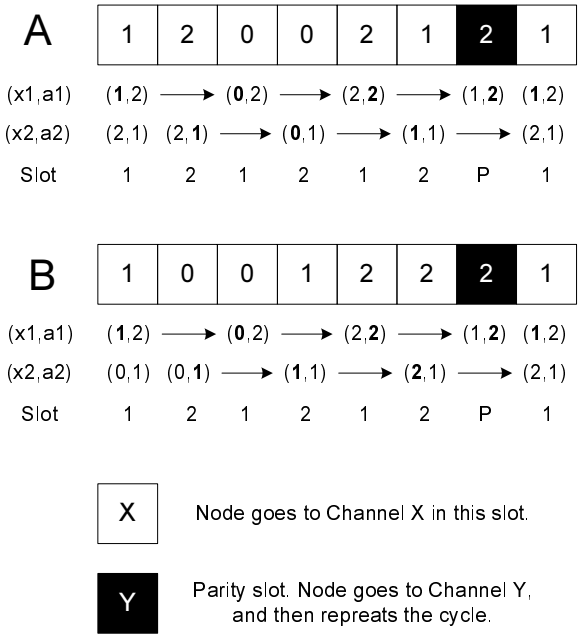


Figure 5.2: An illustration of channel hopping schedules for two nodes A and B with 3 channels and two channel seeds. P denotes the parity slot.

mathematical property that two nodes will guarantee to meet at least once per cycle. Specifically, consider any pair of two channel (x, a) and (x', a'), the following properties hold as shown in [22].

- If $x = x'$ and $a = a'$, they are synchronized with each other.
- If $x = x'$ and $a \neq a'$, they will meet exactly once per cycle.
- If $x \neq x'$ and $a = a'$, they will meet exactly once per cycle at the parity slot.

In the third case above, SSCH has to use parity slots where nodes hop to the channel equal to the seed to prevent the off-schedule of nodes sharing the same seed.

When joining the network, each node will randomly generate a set of channel seeds. Nodes periodically broadcast their schedule, represented by the set of generated channel seeds. For node A wanting to send to node B, A first learns the schedule of node B when they are occasionally in the same channel. Then, A will change one of its channel seeds to one of the channel seeds broadcasted by B. In this way, A will partially synchronize with B. When A is in the same channel with B, it uses CSMA/CA to transmit

the packets to B to minimize the collisions with other senders in the same channel. Also, when A changes the channel seed to synchronize with B for its transmission, A will mark that channel seed as "sending seed". Node B, when seeing a channel seed that always has packets destined to it, will also mark that channel seed as "receiving seed" and *lock* that channel seed. A locked channel seed cannot be changed for any sending purpose. However, if a node wants to send packets and all channel seeds are locked, it will pick a channel seed and change it to the intended receiver's channel seed. In the example above, assuming A wants to send to B, A is partially synchronized with B because their first channel seeds are identical. A will mark its first channel seed (i.e., $(1, 2)$) as "sending seed". B will mark its first channel seed (i.e., $(1, 2)$) as "receiving seed".

5.1.2 Attack Model

There are several compromised nodes that are insider jammers. For a jammer j with the jamming rate p_j^{jam}, in each time slot, if it decides to jam with probability p_j^{jam}, it will uniformly pick a random channel in $\mathcal{C}$ to perform reactive jamming attacks. That means, once it is on the target channel, it will look for preambles and jam the bodies of the data packets. However, if it decides not to jam, it will follow the protocol like other honest nodes.

5.2 Basic Idea: Receiving-based Alibis

The basic idea is to exploit the limited capabilities of the jammer j: he cannot jam and receive simultaneously. This limitation implies that if he jams on one channel in a time slot, he cannot receive in the same time slot. This opens a chance for good nodes to *prove their honesty*. If a group of nodes receives the same corrupted packet content in a time slot, they can coherently show that they cannot be the jammer at that time slot. Such jammed packet contents are referred to as *proof*. Combinations of same proofs in the same time slot will create *receiving-based alibi (R-alibi)*. Nodes that obtain the lowest number of R-alibis are accused as jammers.

5.3 Basic Alibi Protocols

5.3.1 Proof Generation Protocol

When an honest node is idle in any time slots (i.e., no sending or receiving), it switches to a uniformly random channel in $\mathcal{C}$ to become a witness. However, if a node is never idle (i.e., it is always in sending or receiving mode), it has to dedicate several time slots to become a witness. Denote $p_i^{witness}$ to be the probability of a node being witness in each time slot. Intuitively, increasing $p_i^{witness}$ will increase the probability of being a witness and the probability of getting alibis but also decrease its network performance. For example, if a node has always a packet to send, $p_i^{witness} = 0.2$ means it will dedicate 20% of its slots to become a witness. Thus, $p_i^{witness}$ can be used as a parameter to control the trade-off between the probability of getting alibis and the degradation of the network performance.

When a node i becomes a witness in a time slot t on channel $c \in \mathcal{C}$, it receives the packet content PKT_t^c regardless of whether the packet is decodable or not. It will get the hashed content of the received packet by using a good hash function H (e.g. CRC, SHA1 or MD5) and create a proof in the following form:

$$proof = (t, c, H(PKT_t^c)).$$

An honest node stores proofs and sends them to G when it believes that it is under jamming attack. A node detects the presence of a jamming attack when the number of its sending packets getting unacknowledged exceeds a threshold. Once it detects that it is under attack, it will switch to the *proof-exchange* mode in which it tries to send proofs that has been collected to G using a jamming-resistant communication. Depending on the maximum size of the data packet, the node will add as many proofs as possible into the proof packet destined to G. The proof packet will have the following format.

$$m = (i, (i, proof_1, \ldots, proof_{max})_{k_G}).$$

, where $proof_j$ is the proof j-th that node i has collected, max is the maximum number of proofs that can be fit into the data packet and k_G is the public key or shared key of G. The encryption ensures that other nodes receiving the proof packet cannot do the replay attacks and that G can verify the

integrity of the proof packet. If the proof packet is acknowledged by G, the node will move on and send another set of proofs until it has no proofs to send. Otherwise, it keeps trying sending the same proof packet.

Sending the proof to G needs a special handling because the jammers may target to jam the proof packets destined to G. Such jamming attacks on exchanging proof packets are possible because the schedule of G is known to the jammers. Thus, the communication between the nodes and G has be jamming-resistant.

5.3.2 Jamming-resistance for Proof-exchange Protocol

To make the proof-exchange protocol jamming-resistant, we apply a similar technique in the UFH system [15]. During the proof-exchange period, honest nodes randomly pick a channel from $\mathcal{C}$ in each time slot. Note that nodes still use CSMA/CA to send proof packets to reduce collisions. When getting acknowledged from G for a proof packet, a node moves on to the next proof packet. The purpose of the random selection of the channel during the proof-exchange period is to make it harder for the attackers to jam. More importantly, it will prevent the jammers to perform slander attacks on any node because the sending pattern is random and unknown to the attackers.

Because the schedule of G is known to the attackers, it is still possible that attackers specifically target the jamming attacks on node G to block any possible communication to G. To avoid this situation, node G has to randomize its schedule as well. Specifically, node G starts its *proof-exchange* period in which it will randomize its schedule under two conditions: 1) it receives a significant number of corrupted packets and 2) it receives a significant number of distinct nodes sending proof packets to it. The first condition covers the situation where node G is under jamming attack. The second condition covers the situation where the jamming attackers target all other nodes but node G so that G does not start the proof-exchange mode. Both conditions involve thresholds which are system parameters. When switching to proof-exchange period, node G will randomly pick a channel in each time slot. Furthermore, it will only stay in the receiving mode until it can identify the jammers.

5.4 Dealing with Non-colluding Attackers

In this section, we give the analysis of the non-colluding attackers case under the basic alibi protocols proposed in Section 5.3.

Theorem 3 (Identifying non-colluding attackers) *In the network $\mathcal{N}$ with the set of non-colluding jammers $\mathcal{J} = j_1, \ldots, j_{|\mathcal{J}|}$ with the jamming rate $p_{j_1}^{jam}, \ldots, p_{j_{|\mathcal{J}|}}^{jam}$, the R-alibi framework can identify any jammer $j \in \mathcal{J}$ if*

$$p_{rmin}^{witness} \times (1-(1-\frac{p_j^{jam}}{|\mathcal{C}|})(1-a)) \times (1-(1-\frac{1-p_j^{jam}}{|\mathcal{C}|})(\frac{1-b}{1-\frac{p_{rmin}^{witness}}{|\mathcal{C}|}})) > (1-p_j^{jam})ab$$

, where $p_{rmin}^{witness}$ is the minimum probability of being witness of all honest nodes in the network,

$$a = 1 - \prod_{i \in \mathcal{J} \setminus j} (1 - \frac{p_i^{jam}}{|\mathcal{C}|})$$

and

$$b = 1 - \prod_{i \in \mathcal{N} \setminus j} (1 - \frac{p_i^{witness}}{|\mathcal{C}|})$$

.

Proof:

Consider a honest node $r \in \mathcal{N} \setminus \mathcal{J}$ and a jammer $j \in \mathcal{J}$. In any time slot, node r will get an alibi if all following three events happen:

- r becomes a witness
- $A(\mathcal{J})$: at least an attacker in $\mathcal{J}$ jams on the channel r is witnessing
- $B(\mathcal{N} \setminus r)$: at least another node witnesses on the channel r is witnessing.

The first condition happens with probability $p_r^{witness}$. The second condition happens with probability $P[A(\mathcal{J})] = 1 - \prod_{i \in \mathcal{J}} (1 - \frac{p_i^{jam}}{|\mathcal{C}|})$, where $(1 - \frac{p_i^{jam}}{|\mathcal{C}|})$ is the probability that attacker $i \in \mathcal{J}$ does not jam on a particular channel. The third condition happens with probability $P[B(\mathcal{N} \setminus r)] = 1 - \prod_{i \in \mathcal{N} \setminus r} (1 - \frac{p_i^{witness}}{|\mathcal{C}|})$, where $(1 - \frac{p_i^{witness}}{|\mathcal{C}|})$ is the probability that node i does not witness on a particular channel. Thus, in an time slot, node r gets an alibi with the probability

$$p_r^{witness} \times P[A(\mathcal{J})] \times P[B(\mathcal{N} \setminus r)]. \tag{5.1}$$

Similarly, for the attacker j, the probability that it can obtain an R-alibi is

$$p_j^{witness} \times P[A(\mathcal{J}\backslash j)] \times P[B(\mathcal{N}\backslash j)]. \tag{5.2}$$

To have r obtains alibis faster than j, we need the term in Eq. 5.1 to be greater than the term in Eq. 5.2.

$$p_r^{witness} \times P[A(\mathcal{J})] \times P[B(\mathcal{N}\backslash r)] > p_j^{witness} \times P[A(\mathcal{J}\backslash j)] \times P[B(\mathcal{N}\backslash j)].$$

It is obviously best for the attacker j to set $p_j^{witness} = 1 - p_j^{jam}$. Thus, the condition becomes

$$p_r^{witness} \times P[A(\mathcal{J})] \times P[B(\mathcal{N}\backslash r)] > (1 - p_j^{jam}) \times P[A(\mathcal{J}\backslash j)] \times P[B(\mathcal{N}\backslash j)] \tag{5.3}$$

.

We have $P[A(\mathcal{J}\backslash j)] = 1 - \prod_{i \in \mathcal{J}\backslash j}(1 - \frac{p_i^{jam}}{|\mathcal{C}|})$. Thus,

$$\prod_{i \in \mathcal{J}\backslash j} (1 - \frac{p_i^{jam}}{|\mathcal{C}|}) = 1 - P[A(\mathcal{J}\backslash j)]$$

. By multiplying both sides by $(1 - \frac{p_j^{jam}}{|\mathcal{C}|})$, we obtain

$$\prod_{i \in \mathcal{J}} (1 - \frac{p_i^{jam}}{|\mathcal{C}|}) = (1 - \frac{p_j^{jam}}{|\mathcal{C}|}) \times (1 - P[A(\mathcal{J}\backslash j)])$$

Therefore,

$$P[A(\mathcal{J})] = 1 - \prod_{i \in \mathcal{J}} (1 - \frac{p_i^{jam}}{|\mathcal{C}|}) = 1 - (1 - \frac{p_j^{jam}}{|\mathcal{C}|})(1 - P[A(\mathcal{J}\backslash j)]).$$

Denote $a = P[A(\mathcal{J}\backslash j))]$. The condition in Eq. 5.3 becomes

$$p_r^{witness} \times (1 - \frac{p_j^{jam}}{|\mathcal{C}|})(1 - a)) \times P[B(\mathcal{N}\backslash r)] > (1 - p_j^{jam}) \times a \times P[B(\mathcal{N}\backslash j)] \tag{5.4}$$

.

Denote $b = P[B(\mathcal{N}\backslash j)] = 1 - \prod_{i \in \mathcal{N}\backslash j}(1 - \frac{p_i^{witness}}{|\mathcal{C}|}) = 1 - (1 - \frac{p_r^{witness}}{|\mathcal{C}|}) \prod_{i \in \mathcal{N}\backslash\{j,r\}}(1 - \frac{p_i^{witness}}{|\mathcal{C}|})$. Thus,

$$\prod_{i \in \mathcal{N}\backslash\{j,r\}} (1 - \frac{p_i^{witness}}{|\mathcal{C}|}) = \frac{1-b}{1 - \frac{p_r^{witness}}{|\mathcal{C}|}}$$

. By substituting the above term and $1 - p_j^{witness} = p_j^{jam}$ into $P[B(\mathcal{N}\backslash r)]$, we have

$$P[B(\mathcal{N}\backslash r)] = 1 - \prod_{i \in \mathcal{N}\backslash r} (1 - \frac{p_i^{witness}}{|\mathcal{C}|}) = 1 - (1 - \frac{p_j^{witness}}{|\mathcal{C}|}) \prod_{i \in \mathcal{N}\backslash\{j,r\}} (1 - \frac{p_i^{witness}}{|\mathcal{C}|})$$

$$= 1 - (1 - \frac{1 - p_j^{jam}}{|\mathcal{C}|})(\frac{1-b}{1 - \frac{p_r^{witness}}{|\mathcal{C}|}})$$

Thus, the condition in Eq. 5.4 becomes

$$p_r^{witness} \times (1 - (1 - \frac{p_j^{jam}}{|\mathcal{C}|})(1-a)) \times (1 - (1 - \frac{1 - p_j^{jam}}{|\mathcal{C}|})(\frac{1-b}{1 - \frac{p_r^{witness}}{|\mathcal{C}|}})) > (1 - p_j^{jam})ab$$

◇

5.5 Dealing with Colluding Attackers

In this case, we consider a set of colluding jammers $\mathcal{J} = j_1, \ldots, j_{\mathcal{J}}$. Denote $k = |\mathcal{J}|$. *By collusion, we mean attackers can share any information among themselves immediately by any means of communication.* There are several problems when collusion is possible. The first problem is that attackers can coherently lie about their proofs (i.e. hashed content of jammed packets) to create "fake" R-alibis. To cope with this, we require at least $k + 1$ witnesses presenting same hashed content of a jamming packet to create a R-alibi for all witnesses.

The second problem is that *attackers can share alibis.* For example, let us consider the case of 2 colluding attackers. Let us assume one attacker jams the network and the other attacker collects alibis. If there is no alibi-sharing, the jamming one can be detected by our previous proposed detection schemes. However, if alibis are shared to the jamming one, both attackers can get alibis at the rate of other normal nodes and thus cannot be detected.

In what follows, we will discuss how to cope with colluding attackers using the concept of R-chains.

5.5.1 R-chains

Consider an attacker j_1 who jams on channel $c_1 \in \mathcal{C}$ at time slot t. To limit the possibility that j_1 gets a shared alibi from another attacker $j_2(j_2 \in \mathcal{J} \backslash j_1)$ which *correctly* obtains an alibi on channel $c_2(c_2 \in \mathcal{C} \backslash c_1)$ at time slot t, we require j_1 has to be able to *explain its presence* on channel c_2 at time t. If j_1's explanation can be verified, j_1's alibi is valid.

For a node to be able to explain its presence at time slot t on channel $c \in \mathcal{C}$, it has to declare its *sequence of being R-defendant* before time slot t. Let us denote *R-chain(i,s,l)* the sequence of l pairs $(c_1, s)...(c_l, s + l - 1)$ in which node i becomes an R-defendant on channel c_x at time slot $s + x - 1$ $(x = 1..l)$. Thus, *R-chain(i,s,l)* can be used to verify the validity of any R-alibi for node i at any time in between $[s, s + l - 1]$. In other words, node G will only generate an R-alibi for a node i at time slot t on channel c if and only if 1) it receives *R-chain(i,s,l)* before time slot t and 2) the pair (c, t) exists in the chain *R-chain(i,s,l)*.

R-chain can drastically reduce the possibility of alibi-sharing behaviors of the attackers. Essentially, any two attackers j_1 and j_2 can share an R-alibi at time slot t on channel c only when (c, t) exists in both *R-chain(j_1,t_1,l)* and *R-chain(j_2,t_2,l)*. Thus, if all nodes (including attackers) are required to declare their *R-chains* before trying to obtain any R-alibis, the attackers cannot share alibis arbitrarily anymore.

Unfortunately, if R-chain of an honest node is known by the attackers, the node is vulnerable to slander attacks. Basically, attackers can deterministically avoid jamming on channel c at time t if (c, t) is in the R-chains of victim nodes. Thus, victim nodes will not be able to get any R-alibis. To cope with the slander attacks, R-chains need to have certain randomness. This will be discussed in the next section.

5.5.2 One-way R-chains

The basic idea to introduce randomness into an R-chain while still making it verifiable is based on the concept of one-way chains. One-way chains are widely used cryptographic primitive such as in Tesla [25].

A one-way chain is generated based on a one-way hash function F. To generate a one-way chain of length l, we first randomly pick the last element of the chain e_l. Then, we generate the whole chain by repeatedly applying the function F l times (i.e. $e_{l-1} = F(s_l)$, $e_{l-2} = F(e_{l-1})$ and so on). Finally, e_0 is the commitment to the entire one-way chain. e_0 can always be used to verify whether an element belongs to the chain i.e., any e_i belongs to a chain if and only if $F^i(s_i) = e_0$. The chain is released in the order from e_0 to e_l.

There are several key properties of one-way chain that will be used to solve our problem. First, each element e_i in the one-way chain can be considered as a random value uniformly drawn from the output space of one-way hash function F. Second, once the first element of the chain e_0 is released to the network, any later element of the chain e_i $(i > 0)$ cannot be changed and can be verified by checking whether $F^i(e_i) = e_0$. Third, due to the property of one-way hash function F, the knowledge of element e_i does not reveal any information about e_j for any $j > i$. Lastly and most importantly, elements of a one-way chain have to be generated by applying the pre-selected one-way hash function F and cannot be generated arbitrarily.

In our alibi framework, a one-way chain is used to generate a one-way R-chain as follows. Time is divided into epochs of l time slots. An R-chain has a length of l. Each node generates its R-chain at the beginning of each epoch. To generate an R-chain of length l, a node i randomly selects a value s_l^i. The whole chain is then generated from s_l^i by repeatedly applying F in the same way to generate a one-way chain of length l.

For a node i, at a time slot t of an epoch $(1 \leq t \leq l)$, it uses element s_t^i of the chain to calculate the channel c_t^i on which it will become an R-defendant at time slot t. Specifically,

$$c_t^i = s_t^i \mod (\lceil |\mathcal{C}|/p_i^{witness} \rceil)$$

, where $p_i^{witness}$ is the probability of being a R-defendant. So, at time slot t if $c_t^i \leq |\mathcal{C}|$, node i becomes an R-defendant on channel c_t^i and does not

become a witness otherwise. If the hash function is uniform, the probability of being a witness for node i is $\frac{|\mathcal{C}|}{(\lceil |\mathcal{C}|/p_i^{witness} \rceil)} \approx p_i^{witness}$. Thus, this basically emulates the probability $p_i^{witness}$ of being witness of node i. Note that the imprecision of the probability comes from the ceiling operation. However, the imprecision is bounded by

$$\frac{|\mathcal{C}|}{(|\mathcal{C}|/p_i^{witness})} - \frac{|\mathcal{C}|}{(|\mathcal{C}|/p_i^{witness}+1)} = \frac{|\mathcal{C}|}{(|\mathcal{C}|/p_i^{witness}) \times (|\mathcal{C}|/p_i^{witness}+1)} =$$

$$= \frac{p_i^{witness}}{(|\mathcal{C}|/p_i^{witness}+1)} < \frac{1}{|\mathcal{C}|+1}.$$

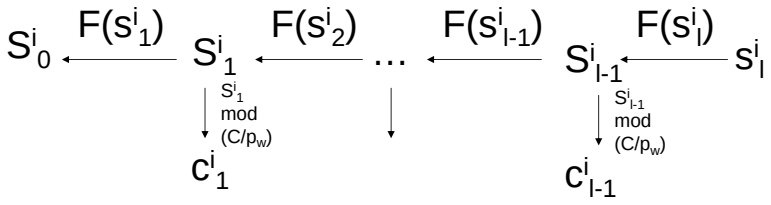


Figure 5.3: An illustration of one-way R-chain

Figure 5.3 illustrates the whole process of generating an one-way R-chain of node i at the beginning of an epoch of l time slots. The chain $s_0^i, \ldots, s_l^i$ is generated in advance. Each s_j^i $(j = 1..l)$ is used to generate c_j^i - the channel that node i will hop to at time slot j in the epoch.

Similar to Section 5.3, the content of a proof of node i at time slot t on channel c now has to include s_t^i. That means, the proof message is

$$m = (t, c, \mathbf{s_0^i}, \mathbf{s_t^i}, H(PKT_t^c)).$$

, where PKT_t^c is the received packet content at time t on channel c.

With this scheme, any recipient (including G) of a proof message of node i can verify whether node i follows its one-way chain by checking whether $F^t(s_t^i) = s_0^i$ and $c_t^i = s_t^i \mod (|\mathcal{C}|/p_w)$. If either check failed, the proof message is invalid and will not be considered for generating alibi.

It is also important to emphasize that by using one-way R-chain, the number of witnesses required to have a valid alibi can be less than $k+1$[1]. This is because the attackers cannot arbitrarily claim or share alibis anymore.

[1](k+1) proofs guarantee a valid alibi because there is k attackers

However, a smaller value of the threshold will have an implication on a *safe-jam strategy* of the attackers presented next.

5.5.3 Analysis of One-way R-chains

As shown in the previous section, because $(s_t^i \mod (\lceil |\mathcal{C}|/p_w \rceil)$ is uniform in $[0, \lceil |\mathcal{C}|/p_w \rceil]$, honest node i still becomes an R-defendant with probability $p_i^{witness}$ and behaves like in the non-colluding scheme.

For attackers, under one-way R-chains, collusion is limited. Specifically, two attackers i and j can collude at the overlaps of their R-chains (i.e. at any time slot t, where $c_t^i = c_t^j$). Thus, if attacker i jams at time t and attacker j becomes an R-defendant on the channel $c_t^j = c_t^i$ at time t, node j will get an R-alibi. Furthermore, if node j shares this alibi to node i, they can achieve both jamming and collecting alibis at the same time. That means such a pair of attackers that colludes in the way just described is undetectable under the one-way R-alibi scheme. We refer to this strategy as *safe-jam strategy*.

The success of the safe-jam strategy depends on the threshold k_{alibi} that the number of witnesses on the same channel at the same time slot has to be greater than to get an R-alibi. The maximum value of k_{alibi} is $(k+1)$ because there are k attackers. Smaller value of k_{alibi} will make both the honest nodes and the attackers to get alibis easier. We now will analyze the safe-jam strategy under the threshold k_{alibi}.

The probability that x attackers select the same particular channel is the binomial distribution with the probability of successful trial $q = 1/|\mathcal{C}|$,

$$P[\text{x attackers select the same channel}] = \binom{|\mathcal{C}|}{x} q^x (1-q)^{|\mathcal{C}|-x}$$

. If $x = k_{alibi}$, the selected channel can be jammed safely. Furthermore, if $x > k_{alibi}$, then the remaining $x - k_{alibi}$ attackers can safely jam any other channels because they already have the shared alibis from the k attackers that stay on the same channel. Therefore, given that $x \geq k_{alibi}$, the number of channels can be jammed safely is $x - k_{alibi} + 1$. Thus, the expected number of channels that can be jammed under the safe-jam strategy is

$$U_{safejam} = \sum_{x=k_{alibi}} P[\text{x attackers select the same channel}] \times (x - k_{alibi} + 1) =$$

$$\sum_{x=k_{alibi}}^{|\mathcal{J}|} \binom{|\mathcal{C}|}{x} q^x (1-q)^{|\mathcal{C}|-x} (x - k_{alibi} + 1) \tag{5.5}$$

.

Figure 5.4 shows $U_{safejam}$, calculated from Eq. 5.5, for the case of $|\mathcal{C}| = 11, 23$ and $k_{alibi} = 2$. The x-axis denotes $U_{safejam}$ - the expected number of channels that can be safely jammed. It is shown that $U_{safejam}$ is around 30% of the total number of channels.

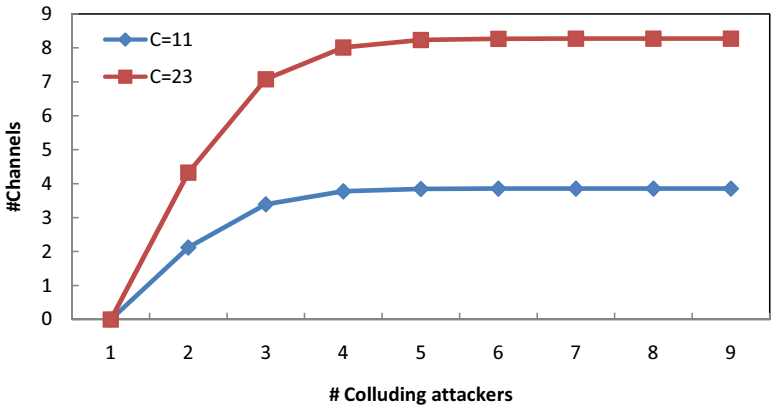


Figure 5.4: Numerical results for the number of jammed channels $U_{safejam}$ under the safe-jam strategy

5.6 Evaluation

5.6.1 Simulation Setup

We evaluate the proposed protocols in TOSSIM. SSCH is implemented by modifying the existing MAC. Each node has 4 channel seeds (similar to the implementation in [22]). Each channel seed can be either "sending", "receiving" or "idle". A node only becomes a witness in the time slots where one of its idle channel seeds is used. There are $n/2$ CBR traffic flows established randomly and uniformly between pairs of nodes. In a CBR flow, the sender will send a data packet to the selected receiver in every $100ms$. In the simulation, all jammers use the same jamming probability p^{jam}. The simulation parameters are listed in Table 5.1. In each scenario, we calculate the average detection probability, average false alarm rate, average detection time and

Parameter	Values
TDMA slot size	100ms
Number of channels	10
Number of nodes n	$[10-40]$
Number of CBR flows	$n/2$
Number of attackers	$[1-9]$
Jamming rate p^{jam}	$[0.1-1.0]$
γ	0.001
Simulation time	200 seconds

Table 5.1: Simulation parameters

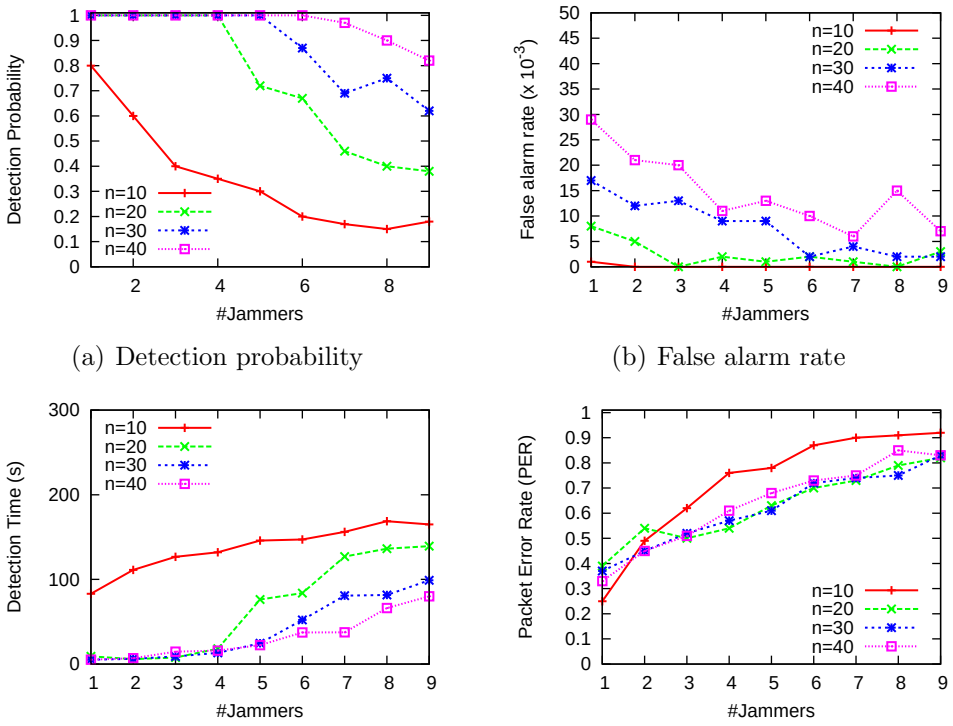
packet error rate. We also repeat each scenario 10 times to get the confident statistics.

5.6.2 Simulation Results

Figures 5.5 and 5.6 show the performance of the proposed system for the case of non-colluding attackers. Specifically, Figure 5.5 shows the results in which the network size is varied from 10 to 40 and the jamming probability of all attackers are set to 0.6. Figure 5.5(a) shows that the detection probability increases when the network size increases. This is because more nodes with more traffic will give create more chances for honest nodes to get alibis. Figure 5.5(a) also shows that more attackers will make it harder to identify them, especially those with low jamming probability. This is because when there are more attackers, any attacker with low jamming probability can have more chances to get alibis from the other attackers' jamming actions. Figure 5.5(b) shows that the false alarm rate is maintained within the expected false alarm rate γ. Figure 5.5(c) further shows the detection time which can be similarly explained as in Figure5.5(a). Figure 5.5(d) shows the average packet loss. As the number of attackers increase, the packet loss rate also increases.

Figure 5.6 shows the results in which the network size n is set to 40 and the jamming probability p^{jam} is varied from 0.1 to 1.0. Figure 5.6(a) shows the detection probability of the proposed system. The detection probability increases when the jamming rate increases. This behavior shows the correctness of the principle of alibis: the more the attackers jam, the easier to detect them. It also shows that more attackers will make it harder to identify them, especially those with low jamming probability. This is because when

there are more attackers, any attacker with low jamming probability can have more chances to get alibis from the other attackers' jamming actions. For attackers with high jamming probability, there will be no difference because they are always busy jamming. Figure 5.5(b) shows the correct false alarm rate according to the threshold γ. Figures 5.6(c) and 5.6(d) further show the detection time and the packet error rate, respectively.



(a) Detection probability

(b) False alarm rate

(c) Detection Time

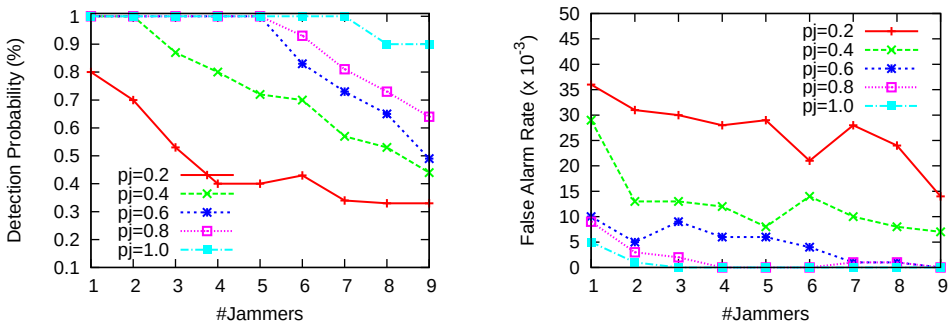
(d) Packet error rate

Figure 5.5: Non-colluding attackers: The impact of network size n ($p^{jam} = 0.6$).

5.7 Discussions

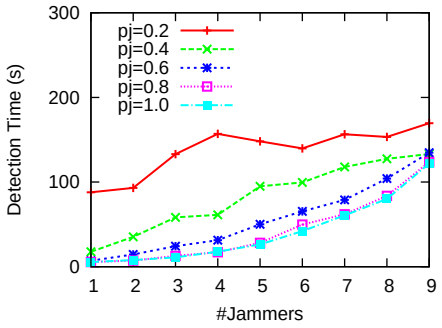
The receiving-based alibi framework discussed in this chapter has the following advantages.

- It can work on multi-channel wireless LANs.
- It can deal with both multiple non-colluding and colluding attackers. With the proposed solution, the damage done by the colluding attackers

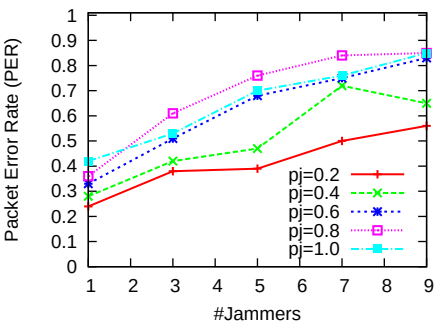


(a) Detection probability

(b) False alarm rate



(c) Detection Time



(d) Packet error rate

Figure 5.6: Non-colluding attackers: The impact of jamming probability p^{jam} ($n = 40$).

is limited.

- It is a reactive defense strategy. That means, it is only activated when the attackers are present in the system. Once the attackers are removed, the alibi scheme will not incur any overhead.

The framework also has following disadvantages. However, it also has the following disadvantages.

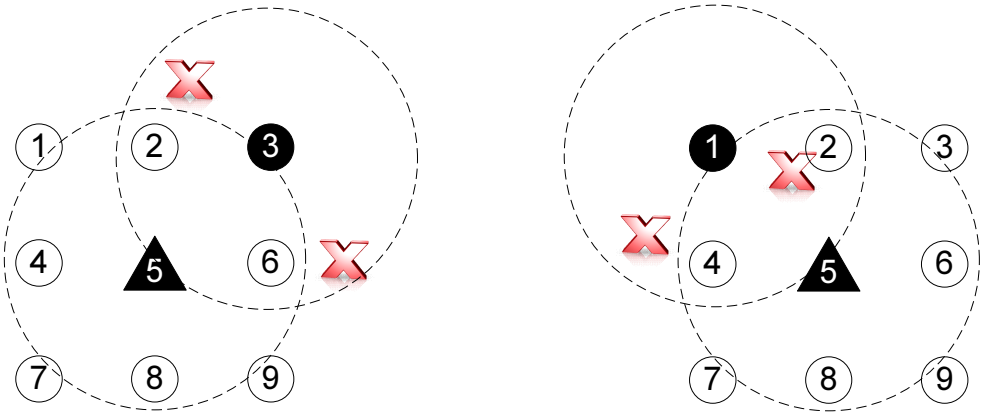
- The colluding attackers can perform a safe-jam strategy to damage the network without getting detected.
- The proposed jamming-resistant communication for proof exchanges may be long, depending on the number of channels.

CHAPTER 6

R-ALIBI DESIGN FOR SINGLE-CHANNEL WLANS

6.1 Overview

In this chapter, we consider a design of the alibi framework for single-channel wireless LANs. Because there is only one channel, sending-based alibis will not possible. Thus, the design relies on the receiving-based alibis. The basic idea is to exploit the half-duplex nature of the jammers: the jammers cannot send and receive at the same time. Thus, whenever an attacker jams a packet, it cannot guess the content of the corrupted packet because it does not know the content of the sending packet. Thus, *any nodes that can show "correct" corrupted (or uncorrupted) packet content can prove that it cannot be the cause of the jammed packet*. The corrupted (or uncorrupted) packet content is "correct" if there are several other receivers agreeing on the same packet content. We call the received packet content as "proof of reception". Combination of proofs of reception from several receivers creates "alibis". Figure 6.1 illustrates how each node gets alibis on jamming events by node 5. In Figure 6.1(a), node 3's message is jammed. Node 2 and 6 receive a corrupted packet. If they both show the content corrupted packet, they can both claim an alibi. Note that nodes 4 and 8 may also receive a packet from node 5. However, they cannot get any alibis because the received packet is uncorrupted. Similarly, in Figure 6.1(b), when node 1's message is jammed, node 2 and 4 will get an alibi by showing proofs of receiving an corrupted packet. Thus, after sufficient jamming events, each honest node in the network gets at least one alibi while the jammer (node 5) cannot get any alibis. At that point, the jammer can be identified.



(a) Node 3's message is jammed by node 5. Nodes 2, 6 get an alibi by showing the content of the corrupted packet.

(b) Node 1's message is jammed by node 5. Nodes 2, 4 get an alibi by showing the content of the corrupted packet.

Figure 6.1: Example of Alibi Scheme.

6.2 Assumptions, Notations and Definitions

6.2.1 Network model

We consider a single-channel WLAN of n nodes. One node is the trusted base station. Denote $\mathcal{N}$ as the set of the nodes in the network (i.e, $|\mathcal{N}| = n$). Each node in the network is equipped with a half-duplex radio, i.e., it cannot transmit and receive at the same time. Thus, there will be non-negligible delay to switch from transmit mode to receive mode and vice versa. We assume CRC-failed packets are still delivered to the upper layer. We assume each node uses CSMA/CA MAC. We also assume a central detection model, i.e., nodes will send information to the base station.

We assume that the messages between the base station and any node are encrypted using the shared key between the base station and that node. Note that the assumption implies that the encrypted message will look uniformly random to any overhearing nodes that do not have the shared key. Also, the encryption/decryption is done at the layer above the MAC layer (e.g., IP layer or application layer) and thus CRC-failed encrypted messages are still delivered to the MAC layer.

6.2.2 Attack model

We assume some nodes in the network are the reactive jamming attackers. Thus, the attackers also have same physical capabilities as other nodes. The attackers, however, have the complete control of the MAC, physical parameters of the radio network interface. The attackers are insider attackers. That means, they are assumed to know any security-related information of the node such as security keys. They also know any protocols used in the system.

The goal of the attackers is to remain undetected while maximizing the number of jammed packets. The attackers use probabilistic reactive jamming strategy. That means whenever an attacker J senses an on-going packet by detecting the presence of a preamble, it will transmit a jamming packet with probability p_J. p_J is called the "reactive jamming probability" and is defined for *each sending packet*. This definition is different from traditional jamming rate in the literature which is defined over *each time slot* regardless of whether there is sending packet in that time slot. Henceforth, the term "jamming rate" refers to "reactive jamming rate" unless explicitly specified.

6.2.3 Definitions and Notations

Denote $\mathcal{S}(t)$ as the set of transmitters at time slot t. Denote $P_{\mathcal{S}\to r}(t)$ as the packet content received by the receiver r under the concurrent sending of senders in $\mathcal{S}(t)$. Denote $PR_{\mathcal{S}\to r}(t)$ ($\mathcal{S} \subset \mathcal{N}\backslash r$) the *proof of reception* for a receiver r at time slot t.

Definition 3 (Alibi of reception (AR)) *An alibi of reception for two receivers r and q at time slot t under the set of sender $\mathcal{S}$ is defined as*

$$AR_{\mathcal{S}\to r,q}(t) = correlation(PR_{\mathcal{S}\to r}(t), PR_{\mathcal{S}\to q}(t))$$

, where correlation is the correlation function defined in Equation 6.1.

Definition 4 (α-alibi neighbors) *Two nodes r and q are called α-alibi neighbors ($0 \leq \alpha \leq 1$) under a set of senders $\mathcal{S}$ in the time slot set $\mathcal{T}$ =*

$t_1, t_2, .., t_{|\mathcal{T}|}$ *if*

$$\mathbb{E}[AR_{\mathcal{S}\to r,q}(\mathcal{T})] = \frac{1}{|\mathcal{T}|}\sum_{\forall t\in\mathcal{T}} AR_{\mathcal{S}\to r,q}(t) \geq \alpha$$

The definition above involves the *average* of alibi correlation of two receivers over a set of time slots. This is because we do not know the exact distribution of the alibi correlation of any two receivers[1]. Thus, our analysis will rely only on the average and deviation of the alibi correlation derived from the experiments described in Section 6.3.

Definition 5 (β-jammer) *A jammer is called β-alibi-jammer ($\beta \geq 0.5$) if it can guess correctly the outcome of the packet from a sender $s \in \mathcal{N}$ caused by its jamming actions with probability β.*

Based on our experiments, which is also confirmed in [26], it is very hard to guess the content of the jammed packet. Thus, the best the jammer can do is a random guess which results in $\beta = 0.5$.

Definition 6 (Complete alibi-safe topology) *A wireless network topology is called (α, κ)-alibi-safe topology if for all pairs of sender $s \in \mathcal{N}$ and jammer $j \in \mathcal{N}$, every node $r \in \mathcal{N}\backslash\{s, j\}$ has at least κ α-alibi neighbors.*

The definition above is strict because it requires that for any possible locations of the jammer and the sender, any receiver always has at least κ α-neighbors. This guarantees that any honest node is always alibi-safe regardless of where the attackers may be. However, if we know some nodes in the network that can be trusted, we can have a looser definition. For example, in our WLAN setting, the base station can be trusted. Thus, we only need to make sure that every node has at least κ α-neighbors under the sending messages from the base station jammed by any possible jammer.

Definition 7 (Alibi-safe topology with trusted senders) *A wireless network topology is called $(\alpha, \kappa, \Lambda)$-alibi-safe topology under a set of trusted senders $\Lambda \subset \mathcal{N}$ if for all pairs of sender $s \in \Lambda$ and jammer $j \in \mathcal{N}\backslash\Lambda$, every receiver $r \in \mathcal{N}\backslash\{j, \Lambda\}$ has at least κ α-alibi neighbors.*

[1]In fact, to the best of our knowledge, there is no theoretical model capturing the distribution of the content correlation under concurrent transmissions.

6.3 Impact of Reactive Jamming Attacks

It might be noticed that in the example above *we assume received corrupted packets, caused by the same jamming event, have the same content* (e.g., nodes 2 and 6 in Figure 6.1(a)). In practice, this might not be the case. Thus, in this section, we will carry out several experiments on a testbed of MICAz motes with CC2420 radio to answer following two questions: 1) *what are the capabilities of reactive jamming attacks?* and 2) *what is the correlation of corrupted packet contents under reactive jamming attacks?*.

6.3.1 Impact of Reactive Jamming Attacks on Network Performance

In our experiments, a reactive jamming attack is performed on a set of 3 nodes as shown in Figure 6.2(a). Nodes are placed such that they can hear each other at the strongest power level (i.e., level 31, 0dbm). S and J are the sender and jammer, respectively. R is the receiver who receives packets from S, J. C acts as the experiment controller.

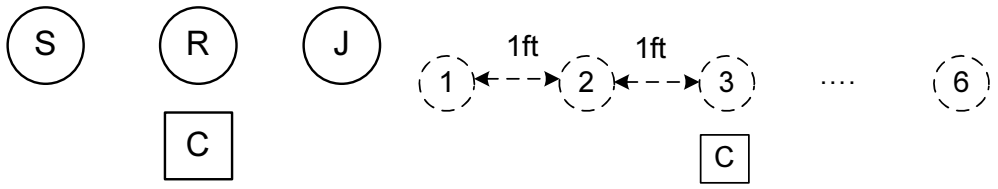
To produce a reactive jamming attack, C will broadcast a message. Upon receiving the broadcast message from C, S starts sending a message of 43 bytes including 32-byte random payload and 11-byte MICAz header. J also starts sending a message of 43 bytes, including 32-byte all-0 payload and 11-byte MICAz header, but with a delay $\delta > 0$. δ is chosen such that the jamming packet will arrive at the receiver after the preamble of the sender. This is just to make sure we have a correct implementation of a reactive jammer. In our experiments, δ is between $150\mu s$ and $200\mu s$. Note that we disable the clear channel assessment (CCA) and backoff mechanism of S and J to ensure concurrent transmissions and CRC check mechanism of R. R records any messages right after the broadcast message from C. The recorded messages are then time-stamped and sent back to C for trace collection. For confident statistics, each experiment is repeated 200 times.

It is known that signal-to-interference-noise-ratio (SINR) will decide the packet content. Theoretically, $SINR = \frac{P_S^R}{P_J^R + P_N}$, where P_S^R and P_J^R are the received powers of the signal sent from S and J at the receiver R; P_N is the noise power. P_S and P_J can be calculated as $P_S^R = P_S \times {d_{SR}}^{-\alpha}$ and $P_S^R = P_S \times {d_{JR}}^{-\alpha}$, where P_S, P_J are the sending powers of S, R; d_{SR}, d_{JR} are

the distance from S, J to R; and α is the path loss factor. Therefore, the major factors affecting SINR are the distance between $S \rightarrow R, J \rightarrow R$ and the sending powers of S, J. Thus, in our experiments, we vary the distance between $S \rightarrow R, J \rightarrow R$ and the sending powers of S, J. We put 6 MICAz motes in the line as shown in Figure 6.2(b). In a reactive jamming attack scenario, we have one sender at mote i ($i = 1..6$) with the sending power level k ($k = 1..30$), one jammer at mote j ($j \neq i, j = 1..6$) with the sending power level l ($l = 1..30$) and 4 receivers at remaining nodes. We try all possible combinations of (i, j) in 6 positions where $j > i$. For each (i, j) pair, we will perform following experiments.

- We measure the received signal strength indication (RSSI) from the sender to each receiver R, denoted as $RSSI_{SR}(d_{SR}, P_S)$, without any sending of the jammer.
- Similarly, we also measure the RSSI from the jammer to each receiver R, denoted as $RSSI_{JR}(d_{JR}, P_J)$, without any sending of the sender.
- We collect corrupted packets at each receiver under reactive jamming attacks (i.e., both sender and jammer sending) for content correlation calculation.

Figure 6.3(a) shows the RSSI at a receiver 1ft away from the sender and the receiver at different sending power levels, i.e., $RSSI_{SR}(1ft, P_S)$ and $RSSI_{JR}(1ft, P_J)$. The x-axis is sending power level of the sender and the jammer (P_S, P_J). The y-axis is the RSSI in dbm.



(a) Reactive jamming attack configuration

(b) Topology of 6 nodes

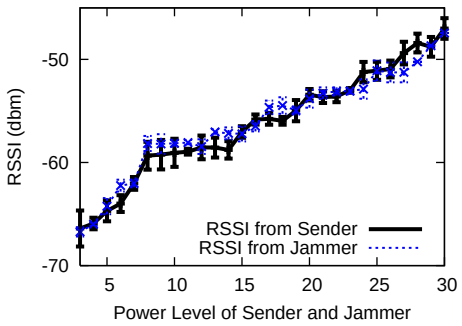
Figure 6.2: Reactive jamming experiment settings

Figure 6.3(b) shows the packet error rate under reactive jamming attacks. The x-axis, denoted by "RSSI by sender" (i.e., $RSSI_{SR}$), is the RSSI of the signal from the sender to the receiver. Note that the RSSI metric takes

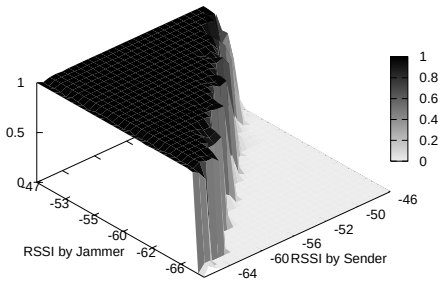
into account of both the sending power and the distance between the sender and the receiver. Similarly, the y-axis, denoted by "RSSI by jammer" (i.e., $RSSI_{JR}$), is the RSSI of the signal from the jammer to the receiver. A pair of RSSI from the sender and the jammer characterizes a receiver. The z-axis shows the packet error rate of each receiver.

From the graph, we have the following observations.

- If $RSSI_{SR}(d_{SR}, P_S) >> RSSI_{JR}(d_{JR}, P_J)$, the packet error rate decreases sharply to 0. This region is referred to as the "white" region. Receivers in this region are called white receivers.
- If $RSSI_{SR}(d_{SR}, P_S) << RSSI_{JR}(d_{JR}, P_J)$, the packet error rate increases sharply to 1. This region is referred to as the "black" region. Receivers in this region are called black receivers.
- When $RSSI_{SR}(d_{SR}, P_S)$ and $RSSI_{JR}(d_{JR}, P_J)$ are close ($< 5dbm$ difference), the packet error rate is between 0 and 1. This region is referred to as the "grey" region. Receivers in this region are called grey receivers.



(a) Power level vs. RSSI



(b) Packet Error Rates (z-axis)

Figure 6.3: Results on reactive jamming experiments

6.3.2 Reception Correlation under Reactive Jamming Attacks

We want to see the correlation[2] of received packet content of any pair of receivers under reactive jamming attacks. We treat a packet content as a

[2]The term "correlation" used in this paper means the similarity.

binary string. The correlation of two packet contents is defined as the similarity of the two corresponding binary strings. The similarity of two binary strings B_1, B_2 of length l is defined as

$$correlation(B_1, B_2) = 1 - \frac{\mathcal{H}(B_1, B_2)}{l} \tag{6.1}$$

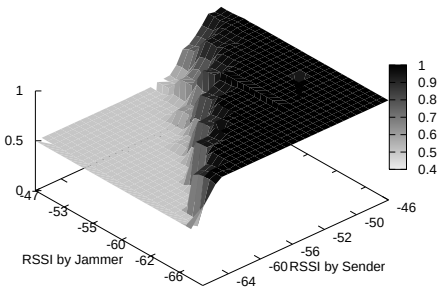
, where $\mathcal{H}(B_1, B_2)$ is the Hamming distance of B_1 and B_2 defined as the number of positions at which the corresponding bits of B_1 and B_2 are different. The correlation has a range of $[0, 1]$. Correlation of 0 means two packet contents are completely different. Correlation of 1 means two packet contents are identical.

We correlate all received packet content of each pair of receivers. Thus, we have a correlation table whose lines are tuples of $(RSSI_{SR_1}, RSSI_{JR_1}$ $, RSSI_{SR_2}, RSSI_{JR_2}, CORR)$. The pairs $(RSSI_{SR_1}, RSSI_{JR_1})$ and $(RSSI_{SR_2},$ $RSSI_{JR_2})$ characterize the first and the second receiver, respectively. $CORR$ is the average of correlation of received packet content of R_1 and R_2. Note that in our experiments, the lowest correlation is 0.5, statistically. This is because the content of the sending packet is uniformly generated, i.e., each bit is uniformly generated between 0 and 1. After constructing the correlation table, we have the following observations.

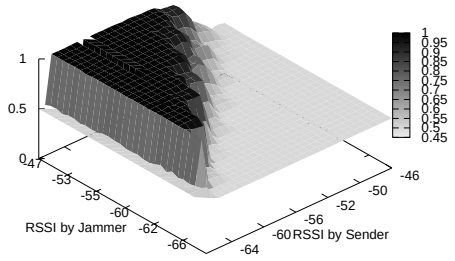
- For a white receiver (i.e., successful reception), it will have very strong correlation (close to 1) with other white receivers. This is obvious because nodes in the white region have successful packet reception. It has a wide range of weak correlation with grey receivers (range of $[0.55, 0.95]$). This can be explained as the grey receivers still have some correct bits from the sender's content. However, a white receiver has very weak correlation with black receivers (range of less than 0.55). Figure 6.4(a) shows the correlation of a typical white receiver R ($RSSI_{SR} = -50dbm$, $RSSI_{JR} = -64dbm$) with all other receivers.

- Similarly, for a black receiver (i.e., unsuccessful reception), it has a very strong correlation with other black receivers, a wide range of weak correlation with grey receivers and very weak correlation with white receivers. Figure 6.4(b) shows the correlation of a typical black receiver R ($RSSI_{SR} = -59dbm$, $RSSI_{JR} = -54dbm$) with all other receivers.

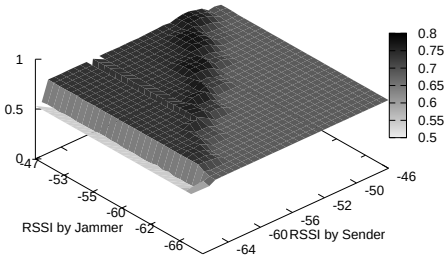
- For a grey receiver, it has a strong correlation (range of $[0.6, 0.8]$) with



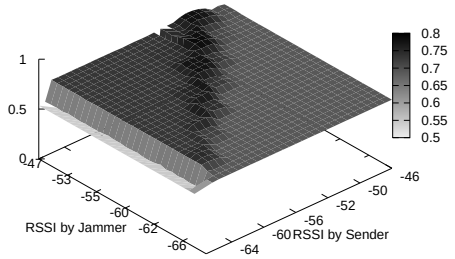
(a) Correlation of white receiver R ($RSSI_{SR} = -52dbm$, $RSSI_{JR} = -64dbm$)



(b) Correlation of black receiver R ($RSSI_{SR} = -59dbm$, $RSSI_{JR} = -54dbm$)



(c) Correlation of grey receiver R_1 ($RSSI_{SR_1} = -59dbm$,$RSSI_{JR_1} = -61dbm$)



(d) Correlation of grey receiver R_2 ($RSSI_{SR_2} = -53dbm$, $RSSI_{JR_2} = -54dbm$)

Figure 6.4: Experiment results on the correlation (z-axis: darker colors represents higher values)

other grey receivers in the grey region. It has wide range of a week correlation with black and white receivers. Figure 6.4(c) shows the correlation of a grey receiver R ($RSSI_{SR} = -59dbm$, $RSSI_{JR} = -61dbm$) with all other receivers.

From this experimental study, we can conclude that **the correlation of received packet content under reactive jamming attacks has the locality property**. That means, receivers closer in the RSSI plane have stronger correlations of packet contents. This is very important for the design of alibi framework.

6.4 Alibi Identification Algorithms

6.4.1 Identifying Non-colluding Attackers

Algorithm: The basic identification algorithm relies on the fact that a jammer can only do a guess with β correlation with the content of the packet it jams. That means, whenever it jams a packet, it only has β correlation while the other nodes get at least α correlation (statistically). Let us define an alibi score function *ascore* for a node r in a time slot set $\mathcal{T}$ as follows

$$ascore_r^{\alpha}(\mathcal{T}) = \sum_{t \in \mathcal{T}} \delta_r^{\alpha}(t) \tag{6.2}$$

, where $\delta_r^{\alpha}(t) = \begin{cases} 1 & \text{if } \max_{q \in \mathcal{N} \backslash r} AR_{S \to r,q}(t) \geq \alpha \\ 0 & \text{otherwise} \end{cases}$

$\delta_r^{\alpha}(t)$ is the alibi indicator function for r at time t. It indicates whether at time slot t, a node r has an alibi correlation greater than α with a node in $\mathcal{N} \backslash r$. Thus, the necessary condition to identify a jammer j is

$$ascore_r^{\alpha}(\mathcal{T}) > ascore_j^{\alpha}(\mathcal{T}), \forall r \in \mathcal{N} \backslash j \tag{6.3}$$

. In other words, a node is accused if it has lowest alibi score.

Theorem 4 (Identifying non-colluding attackers) *In (α, κ)-alibi-safe topology ($\kappa \geq 1$) with a set of β-jammers $\mathcal{J} = j_1, \ldots, j_{|\mathcal{J}|}$ with jamming rate $p_{j_1}^{jam}, \ldots, p_{j_{|\mathcal{J}|}}^{jam}$, the alibi scheme can identify any attacker $j \in \mathcal{J}$ if*

$$p_s^{max} \leq 2 - \frac{1 - p_j^{jam}}{p^{agg}(\mathcal{J})}$$

, where $p^{agg}(\mathcal{J}) = 1 - \prod_{i=1}^{|\mathcal{J}|}(1 - p_{j_i}^{jam})$ is the aggregated jamming rate of the jammer set $\mathcal{J}$ and p_s^{max} is the maximum of sending probabilities of the honest nodes (and thus, $1 - p_s^{max}$ is the minimum witness probability).

Proof: Consider a set of time slots $\mathcal{T} = t_1, ..., t_{|\mathcal{T}|}$ that have sending packets. For each time slot $t \in \mathcal{T}$, a jammer j can increase its alibi point by 1 if it does not jam and at least another jammer jams. This probability is $(1 - p_j^{jam}) \times (1 - \prod_{i=1, i \neq j}^{|\mathcal{J}|}(1 - p_{j_i}^{jam}))$. If j jams at time t, it is obvious that

$\delta_j^\alpha = 0$. Thus, the overall alibi score of j is

$$ascore_j^\alpha(\mathcal{T}) = |\mathcal{T}| \times (1 - p_j^{jam}) \times (1 - \prod_{i=1, i\neq j}^{|\mathcal{J}|} (1 - p_{j_i}^{jam})).$$

Similarly, for the honest node r that has the highest sending probability p_s^{max}, its accumulated alibi score is

$$ascore_r^\alpha(\mathcal{T}) = |\mathcal{T}| \times (1 - p_s^{max}) \times (1 - \prod_{i=1}^{|\mathcal{J}|} (1 - p_{j_i}^{jam})).$$

Substitute to the Inequality 6.3 and do some manipulations, we have

$$p_s^{max} < 1 - (1 - p_j^{jam}) \times \frac{(1 - \prod_{i=1, i\neq j}^{|\mathcal{J}|}(1 - p_{j_i}^{jam}))}{(1 - \prod_{i=1}^{|\mathcal{J}|}(1 - p_{j_i}^{jam}))} = 2 - \frac{1 - p_j^{jam}}{p^{agg}(\mathcal{J})}.$$

$\diamond$.

It is easy to derive following corollaries from Theorem 4.

- If there is one attacker, it will be identified as long as $p_s^{max} < 1$. Note that $p_s^{max} = 1$ means the node always sends in any time slot.
- The above condition does not apply for $p_j^{jam} = 1$. However, if $p_j^{jam} = 1$, it is obvious that j will be identified as long as $p_s^{max} < 1$.
- If $2 - \frac{1-p_j^{jam}}{p^{agg}(\mathcal{J})} \geq 1 > p_S^{max}$, i.e., $p_j^{jam} \geq \frac{\prod_{i=1, i\neq J}^{|\mathcal{J}|}(1-p_{j_i}^{jam})}{1+\prod_{i=1, i\neq J}^{|\mathcal{J}|}(1-p_{j_i}^{jam})}$, j is also identified regardless the value of p_s^{max}. Because $\prod_{i=1, i\neq J}^{|\mathcal{J}|}(1 - p_{j_i}^{jam}) \leq 1$, it is easy to prove that if $p_j > 0.5$, j can also be identified regardless the value of p_s^{max}.
-

It is easy to see that the above identification algorithm also works for alibi-safe topologies with trusted senders. Instead of considering packet sending events from any senders, we only consider the packet sending events from the trusted senders.

6.4.2 Identifying Colluding Attackers

Definition 8 (Colluding attackers) *Colluding attackers are those who have pre-shared knowledge among themselves.*

Note that the definition above allows the attackers to collude through a pre-shared knowledge only. It does not consider the case where attackers can share new knowledge during the network operations (e.g., their proofs of reception). However, collusion via pre-shared knowledge is a very strong attacker model as follows. First, with pre-shared knowledge, colluding attackers can agree on a content of "fake" proofs sent to the detector. In this way, an attacker who jams is still able to get alibi correlations with the other attackers. This collusion will help the jammers to escape from the above alibi identification algorithm. In a bigger picture, with pre-shared knowledge, the attackers can arbitrarily manipulate the alibi correlation among themselves. However, they cannot manipulate the alibi correlation with other honest nodes. Second, with pre-shared knowledge, colluding attackers can coordinate who jams which slot. That means, any two attackers will never jam at the same time slot. In this way, the jammers never waste their jamming effort. Third and last, it is possible to perform a coordinated jamming attack to create "real" proofs as follows. At a time slot, first jammer sends a packet. Second jammer jams the packet of the first sender. Thus, the rest of jammers and honest nodes can all get real proofs and real alibi correlations among each other. Even though the first two jammers might not get any alibi correlation[3], they can get compensated later when other jammers take turn. Apparently, this type of coordinated jamming attack can break any alibi identification algorithms in complete alibi-safe topologies where no senders can be trusted. Thus, we are now going to present an identification algorithm to *identify colluding attackers in alibi-safe topologies with trusted senders*. Note that in WLAN, the set of trusted nodes could be the base station alone.

Algorithm: The basic strategy to deal with colluding attackers is to exploit their only weakness: they cannot manipulate the alibi correlation with other honest nodes under the jammed packets from the trusted senders $S \in \Lambda$. Thus, we define the alibi indicator function for a node r as follows.

[3]In fact, they may be able to get alibi correlation because both the content of sending packet and jamming packet are known.

$$\tilde{\delta}_r^{\alpha}(t) = \begin{cases} 1 & \text{if } q_\kappa \in \Upsilon_r(\mathcal{N}), AR_{S\to r,q_\kappa}(t) \geq \alpha \\ 0 & \text{otherwise} \end{cases}$$

, where $\Upsilon_r(\mathcal{N})$ is the list of nodes in $\mathcal{N}$ decreasingly sorted by the correlation with r at time t and Q_k is the k-th element in $\Upsilon_r(\mathcal{N})$. Once again, the alibi score is defined as

$$\tilde{ascore}_r^{\alpha}(\mathcal{T}) = \sum_{t\in\mathcal{T}} \tilde{\delta}_r^{\alpha}(t)$$

. Similar to the case of non-colluding attackers, an attacker j is identified if

$$\tilde{ascore}_r^{\alpha}(\mathcal{T}) > \tilde{ascore}_j^{\alpha}(\mathcal{T}), \forall r \in \mathcal{N}\backslash j$$

Theorem 5 (Identifying colluding attackers) *In a $(\alpha, \kappa, \Lambda)$-alibi network topology ($\kappa > 1$) with a set of β-alibi colluding jammers $\mathcal{J} = j_1, \ldots, j_{|\mathcal{J}|}$ with the jamming target $p_T^{jam} \leq 1$, the alibi scheme can identify at least one attacker if $p_{send}^{max} < \frac{p_T^{jam}}{|\mathcal{J}|}$, where $p_{max}^{send} = \max_{r\in\mathcal{N}\backslash\mathcal{J}} p_r^{send}$.*

Proof: With the definition of the alibi indicator function, if a jammer j decides to make its alibi correlation with other $(\kappa - 1)$ jammers no less than α, then the κ-th correlation of $\Upsilon_j(\mathcal{N})$ is a correlation with an honest node. If j decides to make its alibi correlation with other κ' jammers ($\kappa' < \kappa - 1$) no less than α, the κ-th correlation of $\Upsilon_j(\mathcal{N})$ is a correlation with another jammer. However, the value of the alibi correlation is less than α and thus $\tilde{\delta}_j^{\alpha} = 0$. Thus, regardless of how a jammer manipulates its alibi correlation with other $(\kappa - 1)$ jammers, its alibi score only increases by 1 if it really has an alibi correlation with an honest node that is no less than α. This will ensure that the collusion will not bring any advantages for the attackers compared to other honest nodes in terms of obtaining alibi scores.

Consider an honest r. In a time slot, it obtains an alibi if the following events happen

- r becomes a witness.
- the channel is jammed.

. The first event happens with probability $1 - p_r^{send}$. The second event happens with probability p_T^{jam}. Thus, r obtains an alibi in a time slot with probability

$$(1 - p_r^{send}) \times p_T^{jam} \tag{6.4}$$

.

Consider a jammer j with the jamming rate p_{jam}^{max} that is highest among all attackers' jamming rates. Intuitively, j will be the one obtain least number of alibis among the attackers due to its highest jamming rate. In a time slot, j obtains an alibi when it becomes a witness and the channel is jammed. Thus, j obtains an alibi in a time slot with probability

$$(1 - p_{jam}^{max}) \times p_T^{jam} \tag{6.5}$$

.

Thus, to satisfy Inequality 6.3, we need

$$(1 - p_r^{send}) \times p_T^{jam} > (1 - p_{jam}^{max}) \times p_T^{jam}$$

. That means,

$$p_r^{send} < p_{max}^{jam} \tag{6.6}$$

.

We need the condition above happens to all honest nodes. Thus,

$$p_{max}^{send} < p_{max}^{jam} \tag{6.7}$$

, where $p_{max}^{send} = \max_{r \in \mathcal{N} \setminus \mathcal{J}} p_r^{send}$.

Given a jamming target p_T^{jam} and a set of $\mathcal{J}$ colluding attackers, it is best for the attackers to equally jam the channel. That means, in each time slot, with probability p_T, the attackers decide to jam and with probability $1 - p_T$ they will not jam. Furthermore, the attackers will jam in a round-robin manner to ensure that each attacker has the same jamming rate and thus has equal chance to obtain alibis. Therefore, the condition in Eq. 6.7 becomes

$$p_{max}^{send} < \frac{p_T^{jam}}{|\mathcal{J}|} \tag{6.8}$$

$\diamond$

6.5 Alibi Protocols

The identification algorithm in Section 6.4 requires the proofs to be present at the detector. That means, every node in the network has to participate in the detection algorithm. Nodes that do not report proofs are at risk of being accused as attackers and receive appropriate reaction from the system such as their removals.

Jamming detection: Because sending proofs incurs overhead to nodes, the base station only collects proofs when there is a jamming attack. To detect the presence of jamming attacks, we use a similar detection techniques proposed in [3]. For the uplink traffic (i.e., from nodes to the base station), a jamming attack is declared if the base station receives a significant number of corrupted packets with strong received signal strength. For the unicast downlink traffic, the base station declares the presence of jamming attacks after getting a significant number of sending packets without receiving acknowledgements.

- *Step 1:* Once the base station detects a jamming attack, it starts a *proof-collection* period of $T_{collect}$ in which it starts to broadcast "test" packets at random intervals. The content of the test packets are uniformly drawn from the message space, i.e., each bit in the content is uniformly drawn from $\{0, 1\}$. The uniform randomness ensures that the attacker can only have a blind guess on the content. Note that each honest node always maintains proofs in the latest window interval of $T_{collect}$. At the end of the proof-collection period, the base station broadcasts a *proof-exchange* message.

- *Step 2:* When nodes receive the *proof-exchange* message from the base station, they immediately start sending the proofs that they have just collected in the last $T_{collect}$ time interval to the base station.

- *Step 3:* After receiving proofs from nodes collected in the last $T_{collect}$ time interval, the base station starts the identification algorithm based on the set of time slots in which it sent "test" packets. It removes any identified attackers. After that if there is still jamming attack going on, it repeats Step 1.

All messages between nodes and the base station are transmitted using

BBC-based timing channel. The BBC-based timing communication relies on the timing pattern of sending packets to convey the actual message. It has a strong resistance to reactive jamming attacks and allows concurrent transmissions of multiple senders. One major disadvantage of this timing channel is its low throughput. More details are discussed in Section 6.6.

To cope with the low throughput nature of the BBC-based jamming-resistant timing-channel on which proofs are exchanged, the alibi framework compresses the proofs using a hashing technique called "similarity preserving hashing" (or locality sensitive hashing). Unlike other hashing techniques such as MD5 or SHA-1, similarity hash functions have a special property that the Hamming distance of hash values of two objects is proportional to their original Hamming distance. Therefore, instead of storing and exchanging the raw packet content, each node only needs to keep the hash version of proofs to reduce the storage and communication overhead. Section 6.7 will give more details.

6.6 BBC-based Timing Channel

In the framework, we need a jamming-resistant timing channel between the nodes and the base station. We use the technique mentioned in Section 3.5.1.

BBC is a keyless jamming-resistant broadcast communication proposed in [19]. The basic idea is to have the sender create *"indelible" marks* in an additive-OR channel to convey a sending message. The receiver receives a "packet" containing all the marks and decodes the original sending message. Indelible marks in the additive-OR channel have an important property: the jammers cannot erase the existence of the marks in the channel; they can only create extra marks. Therefore, a received BBC packet may contain more marks than those created by the senders. In other words, the set of marks in the received BBC packet is the super-set of the marks created by senders. Thus, the coding scheme is called *concurrent code* and BBC code is the only known implementation of the concurrent code.

There are several ways to create an additive-OR channel [19]. In the alibi framework, we build a pulse-based channel from the data channel. A pulse is 4-byte preamble-only packet. Multiple pulses sent at the same time results in only one single pulse. In this way, this pulse-based channel is

an additive-OR channel. For BBC code, a pulse is also an indelible mark. Reactive jammers cannot erase marks because they jam only after sensing the preambles[4]. In addition, concurrent sending of multiple marks will result in only one mark. Therefore, the BBC-based timing channel built in this way is not only robust to the reactive jammers but also allows multiple concurrent message transmissions.

A message M of length n bits is encoded into a message M' of length $m = ne$, where e is the expansion factor ($e > 1$). Denote $\Pi(m, i)$ the first i bits of message m and a hash function H (e.g., MD5 or SHA1). For each $i \in (1, n)$, we calculate the location $L(i)$ of the i-th mark corresponding to bit i as $L(i) = H(\Pi(m, i)) \mod m$. Therefore, the message M is encoded into n marks whose locations are from $0..m$. The encoded message M' will then be transmitted in m slots. The marked slots correspond to transmissions of a pulse (i.e., a preamble-only packet). The unmarked slot is equivalent to no transmission.

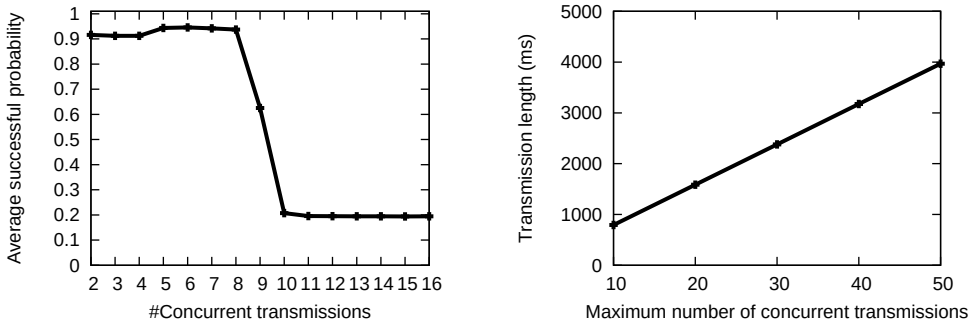
The receiver receives a packet of length m containing at least n marks. The packet may have more than n marks due to either multiple messages sent or extra marks by jammers. The decoding process guarantees to decode any messages contained in the packet. It maintains a tree of maximum depth n rooted by an empty message. At the depth level 1, if the location at ($H(0)$ mod m) is marked, 0 is added into the tree. Similarly, if the location of ($H(1)$ mod m) is marked, 1 is added into the tree. Recursively, at each node at level i with the corresponding message of length i M_i, if ($H(M_i b) \mod m$) ($b = 0, 1$) is marked in the packet, b is added as a child of the node. The tree keeps growing until the level n. Any nodes at level n correspond to decoded messages of length n.

Even though the decoding process guarantees to decode any messages contained in the packet, it also decodes extra messages, referred to as "hallucination". To control the number of hallucination messages, a checksum sequence containing l 0-bits is added into the original message. Furthermore, to control the width of the decoding tree, s bits are inserted in every f bits in the original message. We leave the detailed analysis of these parameters to the original paper [19].

[4]In fact, even if the attackers jam the preambles, it is still possible to detect the jammed preambles. However, the preamble detection has to rely on the energy detection rather than the preamble signature.

It is important to emphasize that there is a critical threshold on the number of concurrent transmissions below which all concurrent transmission are highly successful and above which no concurrent transmission are successful. This critical threshold implies a trade-off between the maximum number of successful concurrent transmissions and the transmission length of the messages. Specifically, to have n nodes sending a message of b bits concurrently with high successful rate, the transmission length has to be $O(n \times b)$. Figure 6.5(a) shows the performance of the BBC-based timing channel in which the maximum number of concurrent transmissions is targeted at 10. As shown in the Figure 6.5(a), there is a sharp decrease of the successful transmission probability (y-axis) from 0.95 to 0.2 when the number of concurrent transmission (x-axis) approaches 10 and goes beyond. Figure 6.5(b) further shows the transmission length (y-axis) versus the targeted maximum number of concurrent transmissions (x-axis).

It is important to emphasize that the proof transmission using BBC in Step 2 of the alibi protocol is not encrypted. If in Step 2, a jammer decides to listen for some proofs, which takes a significant time, and start the proof transmission to the BS, the start of its proof transmission period will be significantly late compared to any other nodes and can easily be detected by the BS.



(a) The critical threshold of BBC for 10 nodes

(b) Maximum no. concurrent transmissions vs. Transmission length

Figure 6.5: Performance of the BBC-based timing channel

6.7 Similarity Hashing (SimHash) & Alibi

Locality Sensitive Hashing (LSH) is a popular technique used in information retrieval to detect near-duplicate documents [27]. Essentially, LSH is a method of performing probabilistic dimension reduction of high-dimensional data. The basic idea is to hash high-dimension input objects so that similar objects are mapped into the same buckets with high probability. In other words, input objects are hashed such that their similarities are preserved in the hash space. This similarity-preserving property is completely opposite to normal hashing techniques (e.g., SHA1 or MD5), where a small difference of inputs might lead to a completely different hash outputs. Formally, a locality sensitive hashing scheme is a distribution on a family $\mathcal{F}$ operating on a collection of objects, such that for two objects x, y,

$$\mathbf{Pr}_{h\in\mathcal{F}}[h(x) = h(y)] = similarity(x, y)$$

, where $similarity(x, y) \in [0, 1]$ is the similarity function defined on the collection of objects.

In the alibi framework, a similarity hashing function $\mathcal{F}$ is used to hash the packet content. That means, the correlation of two packet content B_1, B_2 will be calculated as $correlation(B_1, B_2) = 1 - \frac{\mathcal{H}(\mathcal{F}(B_1),\mathcal{F}(B_2))}{l}$, where $\mathcal{F}(B_1), \mathcal{F}(B_2)$ is the similarity hash of the packet content B_1, B_2.

There are several techniques to implement $\mathcal{F}$ such as random sampling [28], min-wise independent permutation [29], random projection [27] or stable distribution [30]. In the alibi framework, we choose the random projection technique, also referred to as *simhash*, proposed by Charikar [27] due to its simple and efficient implementation [31]. The random projection method of the similarity preserving hashing is designed to approximate the cosine distance between vectors. The basic idea of this technique is to choose a random hyperplane (defined by a normal unit vector $\vec{r}$) at the outset and use the hyperplane to hash input vectors. Formally, we let $h(\vec{v}) = sgn(\vec{v} \cdot \vec{r})$, i.e., $h(\vec{v}) = \pm 1$ depending on which side of the hyperplane $\vec{v}$ lies. Therefore, each possible choice of r results in a binary output. If we choose the vector $\vec{r}$ t times uniformly, we will have a t-bit output of vector $\vec{v}$. For any two vectors $\vec{u}, \vec{v}$, the bits of two t-bit outputs match with probability proportional to the cosine of the angle between them. In [27], the author proves that

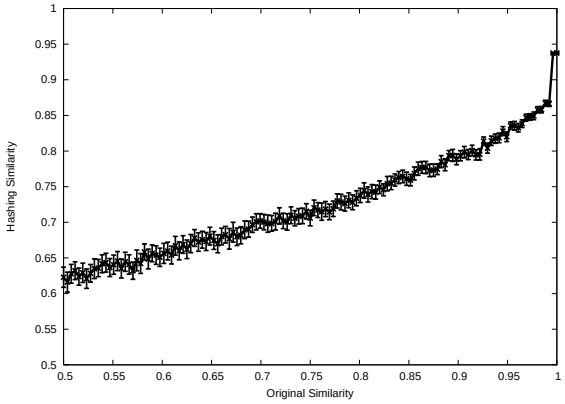


Figure 6.6: Accuracy of Simhash

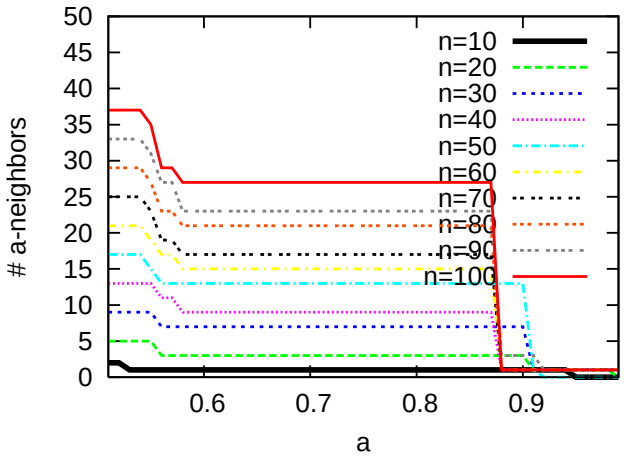
$\mathbf{Pr}[h_r(\vec{u}) = h_r(\vec{v})] = 1 - \frac{\theta(\vec{u},\vec{v})}{\pi}$, where θ is the angle of vectors $\vec{u}$ and $\vec{v}$.

The point of using *simhash* is to reduce the amount of data needed for storing and transmitting proofs. For example, a 32-byte packet content can be sim-hashed into a 2-byte fingerprint while still preserving reasonable accuracy of the similarity with other packet content. Figure 6.6 shows the accuracy of 2-byte sim-hash for a 32-byte vector. Two 32-byte vectors sampled uniformly are sim-hashed. The plot shows the correlation of original 32-byte vectors (i.e., X-axis) and the correlation of the 2-byte sim-hash values (i.e., Y-axis).

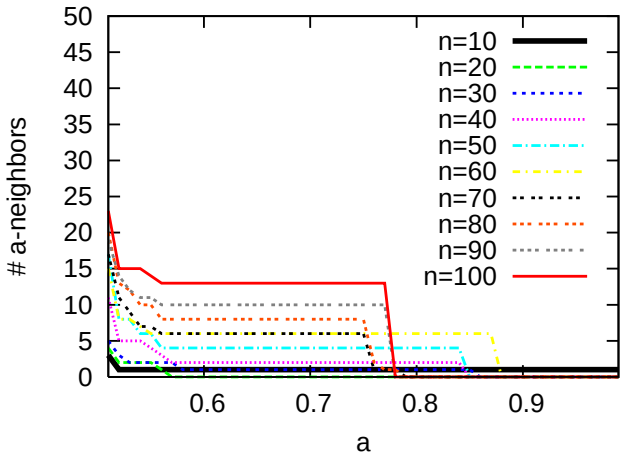
6.8 Evaluation

6.8.1 Evaluation of Alibi-safe Topologies

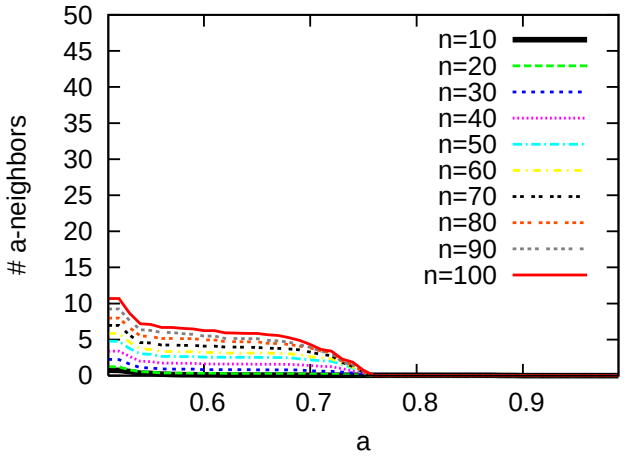
There is a strong connection between alibi-safe topologies and physical topologies. Thus, it is important to determine whether a given physical topology is an alibi-safe topology. Because alibi-safe topology only relies on the received signal strength, we need to assume a propagation model in order to calculate RSSI from the sending power. In this evaluation, we assume a log-normal shadowing path-loss model. In the log-normal shadowing path-loss model, $P_R = P_S - PL(d)$, where $PL(d) = PL(d_0) + 10\alpha_{pl} \log_{10} \frac{d}{d_0} + N_\sigma$ is the path-loss of the radio over a distance d; P_R is the power of the signal at the receiver; P_S is the power of signal at the sender; d_0 is the reference distance and $PL(d_0)$ is the power decay for this distance; α_{pl} is the signal



(a) Star topology



(b) Grid topology



(c) Random topology

Figure 6.7: Physical topologies vs. Alibi-safe topologies

decay factor. N_σ is the zero-mean Gaussian (in db) with standard deviation σ representing the multi-path effects.

To test whether a given physical topology is an (α, κ)-alibi-safe topology under a set of senders Λ, we perform following steps. First, for each sender S in Λ, we calculate the RSSI at each receiver for the message sent by S. We use the correlation table constructed in the experiments described in Section 6.3 to determine whether two receivers are α-neighbor. Then, for each receiver, we count the number of α-neighbors. If any receiver has less than κ α-neighbors under the sender S, the topology is not (α, κ)-alibi-safe topology.

We perform the alibi-safe test for three types of physical topology in a $20m \times 20m$ square: star topology, grid topology and random topology where the set Λ only has the trusted base station located at the center of the square. Figure 6.7 shows the results of the alibi-safe tests for the network size from 10 to 100. The x-axis is the α value and the y-axis is the *minimum* number of α-neighbors for every node (i.e κ). That means, each point represents a possible (α, κ)-alibi-safe topology. For star topology where nodes surround the base station, there is high chance for a node to have alibi neighbors as shown in Figure 6.7(a). For grid topology, it is less alibi-safe as shown in Figure 6.7(b) because nodes have different distance to the base station. The alibi-safe of random topologies really depends on network density as shown in Figure 6.7(c).

In terms of robustness to colluding attackers, star topologies are the strongest. In a star topology, each node has roughly 30% of other nodes as its 0.55-neighbors. For grid topologies and random topologies, these numbers are around 15% and 7%, respectively.

6.8.2 Evaluation of alibi on TOSSIM Simulator

We use TOSSIM as our network simulator for a large-scale evaluation of the alibi framework. TOSSIM does not assume any radio propagation model; instead, it provides a radio abstraction between two-node communication. Specifically, it derives the packet error rate based on the empirical RSSI. Furthermore, it also has a more accurate noise generator using Close Pattern Matching (CPM) algorithm on a given noise traces. In our simulation, the

Parameter	Values
Number of nodes	$n = [10 - 40]$
Number of attackers	$[1 - 9]$
Jamming rate	$[0.1 - 1.0]$
Value for α-neighbors	$\alpha = 0.55$
γ	0.05
Simulation time	1000 seconds
BBC maximum number of concurrent transmissions	50
BBC message length in time	4s
Number of BBC messages per proof-exchange period	10
Number of bits for simhash	16 bits

Table 6.1: Simulation parameters

noise trace is the one obtained from the experiments described in Section 6.3. Because TOSSIM does not provide the correlation of packet contents under reactive jamming attacks, we add that feature at the physical layer using the correlation table obtained from the experiments in Section 6.3. BBC-based timing channel is implemented at the MAC layer in TOSSIM and is transparent to the application layer.

There are two alibi protocols: *the Omniscient protocol (OMS)* and *the Alibi-BBC-Simhash protocol (ABS)*. The Omniscient protocol is the optimal protocol because it assumes every proof is available immediately at the central detector right after its creation. The ABS protocol uses the BBC-based timing channel and simhash.

We evaluate the two alibi protocols under three types of topologies: star topology, grid topology and random topology in a square of $20m \times 20m$ and two types of attackers: non-colluding and colluding attackers. We vary the simulation parameters as shown in Table 6.1. We obtain the detection accuracy (i.e., the detection probability and the false alarm rate), the detection time, the network performance and the network overhead. However, due to the space limit, we only show the results for the star topology.

Detection accuracy: Figure 6.8 shows the detection accuracy of the ABS protocol under non-colluding jamming attacks and colluding jamming attacks. Specifically, Figures 6.8(a) and 6.8(b) show the detection accuracy for non-colluding attackers using the same jamming rate of 0.2. It is shown that as the number of attackers increases, the detection probability decreases. This is because when the network has more attackers, the chance for an attacker to get alibis by the jamming actions of the other attackers also in-

crease. It is shown that as the network size increases from 10 to 40, the detection probability increases. This is because the larger network size increases the number of α-neighbors and makes the attackers busier to jam. The figures also show that the false alarm rate remains similar under different network sizes and number of attackers. This shows the expected behavior of the proposed outlier detection technique. We also obtain the results for different γ but do not plot them here. Essentially, larger value of γ will lead to lower false alarms but also lower detection probability. Figures 6.8(c) and 6.8(d) show the detection accuracy for colluding attackers who collude and coordinate to target at 100% jamming rate. The trends are similar to the case of the non-colluding case. However, the detection probability is slightly worse compared to the non-colluding cases. This is due to the fact that 1) colluding jammers need to jam less than non-colluding jammers for the same aggregated jamming target and 2) each node needs more α-neighbors than the non-colluding case.

Figure 6.9 shows the detection performance of ABS protocol against the OMS protocol for $n = 40$. ABS protocol has a gap of around $0.2 - 0.3$ to the OMS protocol. This is because the nodes cannot send all the proofs to the central detector due to the low throughput of the timing channel.

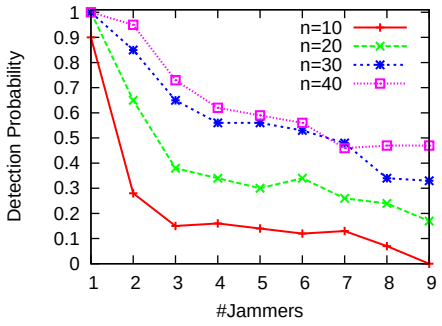
Packet error rate: Figure 6.10 shows the packet error rate of the non-colluding and colluding attackers. As shown in Figure 6.10, with the same jamming rate of 0.2, having more non-colluding attackers only increases the packet error rate sub-linearly. In contrast, colluding attackers can coordinate to achieve 100% jamming. This shows the danger of colluding attackers over the non-colluding attackers.

Detection Time: Figure 6.11 shows the detection time for the case of non-colluding and colluding attackers. As shown in the figures, as the number of jammer increases, it takes longer time to identify them. The explanation is similar to the detection probability.

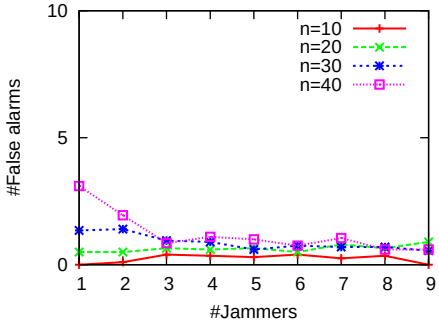
6.9 Discussions

The proposed design of the receiving-based alibi framework has the following advantages.

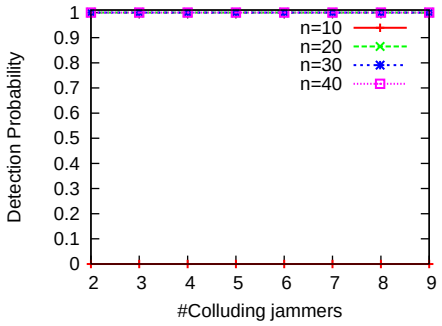
- It can work on current CSMA/CA wireless LANs.



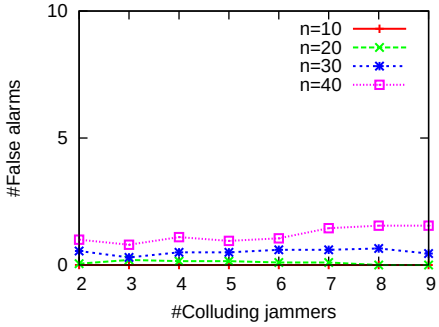
(a) Detection probability (non-colluding)



(b) False alarms (non-colluding)

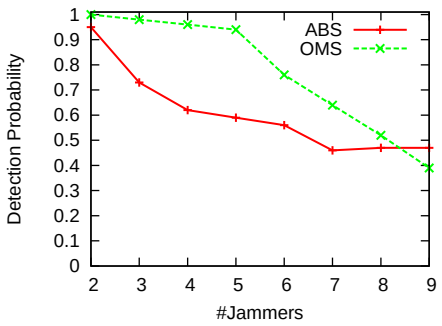


(c) Detection probability (colluding)

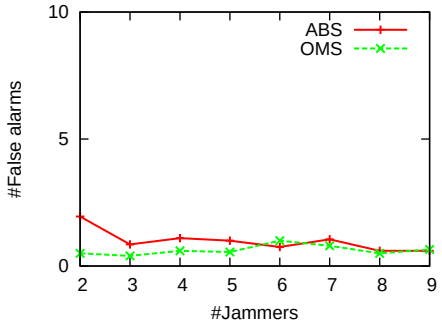


(d) False alarms (colluding)

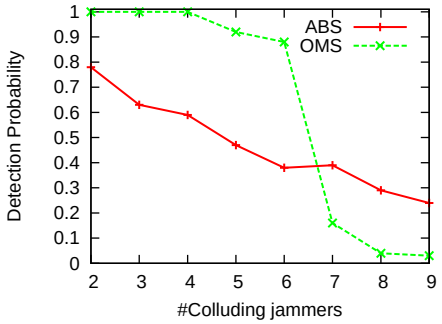
Figure 6.8: Detection accuracy of ABS protocol



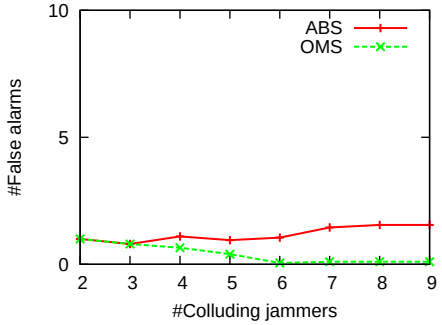
(a) Detection probability (non-colluding)



(b) False alarms (non-colluding)

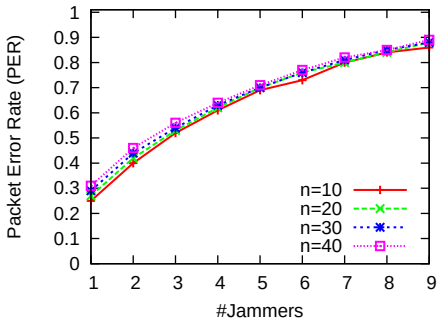


(c) Detection probability (colluding)

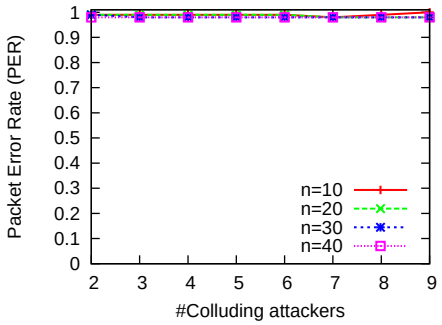


(d) False alarms (colluding)

Figure 6.9: Detection Accuracy: ABS vs. OMS (n=40)

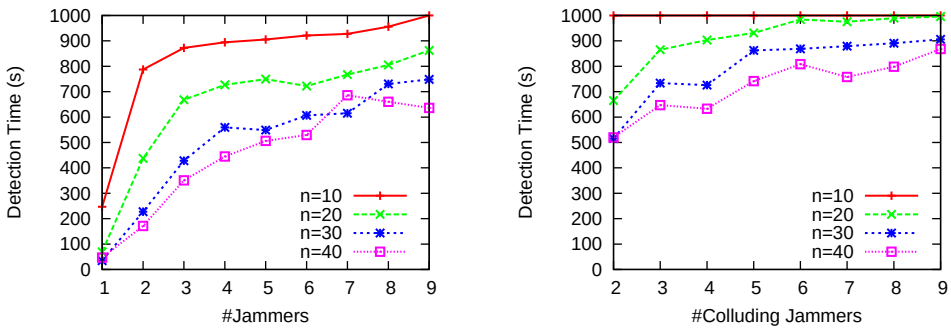


(a) Non-colluding attackers



(b) Colluding attackers

Figure 6.10: Packer error rate of the ABS protocol



(a) Non-colluding attackers (b) Colluding attackers

Figure 6.11: Detection time of ABS protocol

- It can deal with both multiple non-colluding and colluding attackers.
- It is a reactive defense strategy. That means, it is only activated when the attackers are present in the system. Once the attackers are removed, the alibi scheme will not incur any overhead.

However, it also has the following disadvantages.

- The current design of the framework cannot deal with the attackers which can exchange the information during the network operation.
- It cannot defend against the hybrid attacks of colluding attacks and sybil attacks in which one attacker sends the proof on behalf of the other jamming attacker.

CHAPTER 7

RELATED WORK

7.1 Jamming Attacks and Defenses

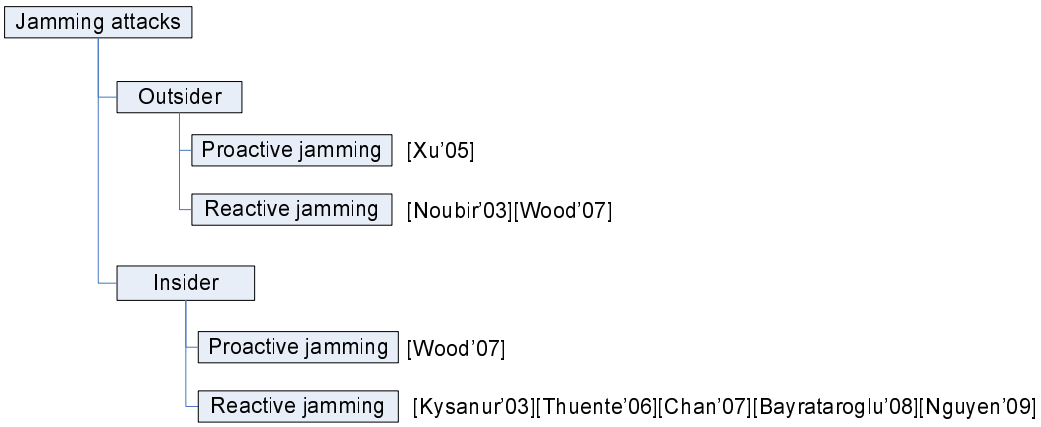


Figure 7.1: Ontology of jamming attacks

There has been plethora body of research work on jamming attacks [32] [2, 7, 33, 10, 34, 35, 36, 37, 38, 39, 40, 41, 42, 9, 43] . Figure 7.1 shows an ontology of jamming attacks with representative works. Jamming attacks can be classified as proactive or reactive. In the proactive jamming strategy, the attacker jams the channel without caring about the on-going communication. A typical example of this type is the continuous jamming [33, 2]. This strategy is the simplest way to perform a jamming attack. However, it is not energy-efficient due to the continuous jamming activity. This also makes the attacker easy to detect. Reactive jamming strategy [38, 39, 40, 41, 42, 9, 43, 2, 7], in contrast, avoids these drawbacks by intelligently listening and jamming the channel. In this strategy, the attacker keeps listening and only jams "important" packets such as control packets [43][44]. Corrupted control packets can drastically reduce the effective throughput of the communication channel [43][44]. The reactive jamming attack is usually

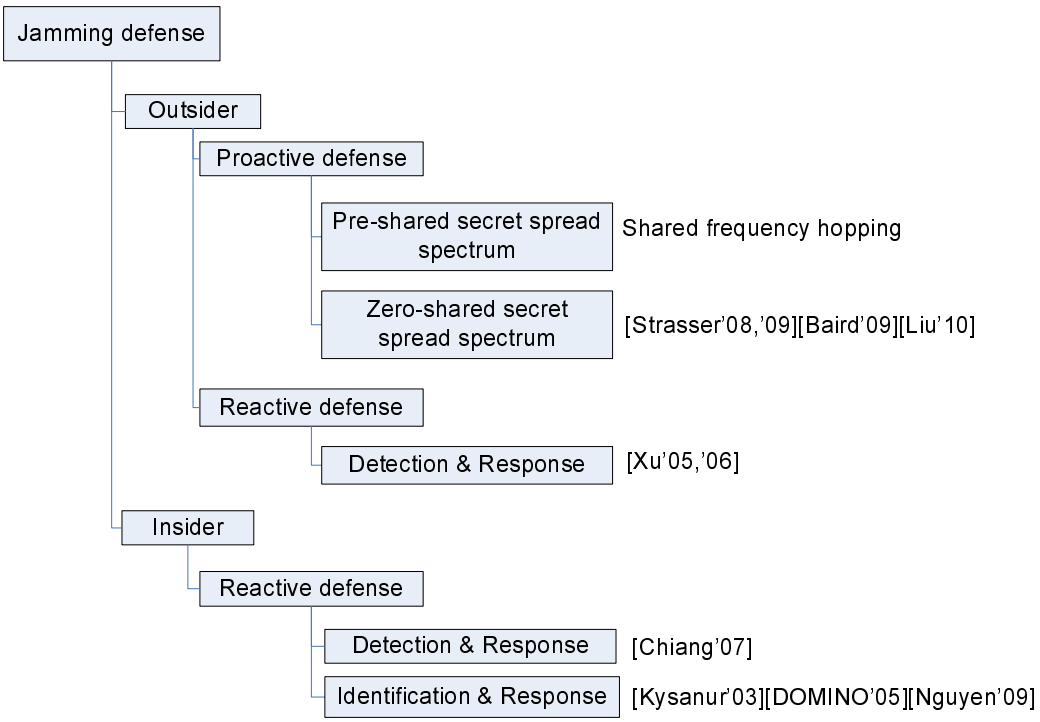


Figure 7.2: Ontology of jamming defenses

more complicated than the proactive jamming attack due to the stealthy nature of the attacker.

Due to the dangers of various jamming attacks, jamming defenses have gained much attention from researchers [45, 46, 47, 48, 49]. Figure 7.2 shows an ontology of jamming attacks with representative works. One of the most effective jamming mitigation is the spread spectrum technique. By hopping the carrier frequency (frequency-hopping spread spectrum - FHSS) or spreading its signal in time (direct-sequence spread spectrum - DSSS), the network can force the jammer to spend several-fold more power than if spread spectrum were not used [50][7]. However, spread spectrum does not work if the jammer knows the hopping-pattern (HP) of the FHSS or the pseudo-noise chip (PN) sequence of DSSS. Once the attacker knows such knowledge, he can jam the channel very effectively. For example, in 802.11 DSSS the PN is a common knowledge and the attacker can easily obtain it [8]. By just using the COTS 802.11 cards, the attacker can easily modify the firmware to have an effective 802.11 jammer [2]. That said, the "outsider" attack (i.e., no knowledge of the HP or PN) can be defended effectively with spread spectrum technology while the "insider" attack is still a problem.

Indeed, dealing with the insider jamming attacks, where the "shared secret" such as shared HP or PN is compromised, is a challenging problem. This problem exists not only in the spread spectrum technology but also in other wireless technologies such as Ultra-wide band (UWB) (pulse-pattern as the shared secret)[50][19]. Unfortunately, there have been few research results on this topic. These research results share the view of considering shared secret as a type of "shared key" among all nodes. From this point of view, dealing with a compromised shared key is similar to the key management in the traditional security literature. Specifically, hierarchical key management and asymmetrical key schemes have been explored in [5] and [19]. In [5], the authors extend the idea from the well-known hierarchical key management to eliminate the compromised shared secret. However, this scheme is designed only for the wireless broadcast network where the base station can send/receive on different channels at the same time. In [19], the authors propose a concurrent coding scheme to form a communication primitive under jamming condition. This can be used as a way to setup a shared key from the asymmetric key by using some techniques like Diffie-Hellman [51]. This scheme, however, is only applicable for point-to-point communication.

7.2 Detecting Mis-behaving/Compromised Nodes

There has been also work related to detecting mis-behaving nodes in the network. In many ways, insider jammers can also be considered as malicious nodes. For example, a misbehaving node may intentionally to send packets to collide with legitimate packets from other nodes. Pioneer work on this topic was first proposed in [44]. After that, there has been many follow-up works on this topic [52, 53, 54, 55, 56, 57]. Essentially, the literature can be classified into two categories: 1) detecting mis-behaving nodes and applying certain penalties to the mis-behaving nodes, and 2) preventing/discouraging mis-behaving nodes by modifying the existing MAC protocols or proposing new design of MAC protocols.

Detecting mis-behaving nodes is possible with the cooperation from honest nodes. The first set of detection schemes provides solutions based on the modification of existing IEEE 802.11. For example, in [44], the proposed

scheme assumes a trusted receiver that assigns back-off value to nodes, or a negotiation of the back-off value among the nodes [52]. Because the back-off values are known to the detector, the mis-behaviors are easy to detect.

In the second set of detection schemes, the detection is done without any modifications to the existing MAC protocols. These schemes are more practical to deploy. However, the detection is much more challenging because each node selects the back-off values independently. Thus, the detector cannot determine with complete certainty whether the deviating behaviors of a node are caused by chances or by real deviations. In DOMINO [54], the detection algorithm estimates the average back-off time, and raise an alarm if the estimation is suspiciously low. In [55], the authors propose to use the inter-delivery time distribution instead of the back-off times of the nodes. By using the Sequential Probability Ratio Test (SPRT), the authors estimate a normal inter-delivery distribution and an attack inter-delivery distribution. A more robust SPRT approach was proposed in [53]. The scheme basically addresses the detection of adaptive intelligent cheaters by using the min-max robust detection framework.

When honest noes behave selfishly, preventing/discouraging mis-behaving nodes can be done by game-theoretic ideas [58][59][60]. In the game-theoretic settings, nodes are assumed to be selfish and try to maximize their access to the medium. The goal of the protocol is to motivate users to achieve a Nash equilibrium in which no nodes want to deviate from the protocol. However, due to the assumption that all nodes are willing to deviate from the protocol, the throughput achieved is substantially less than in the protocols where honest nodes cooperate in the design.

7.3 System-level Fault Diagnosis

System-level fault diagnosis has been an active research area after the pioneer work by Preparata et al. [61], referred to as PMC model. The fundamental assumption in system-level diagnosis is that one node may be able to test single or multiple node(s). The outcome of a test can be either "faulty" or "fault-free". Specifically, a fault-free node can correctly diagnose another node as fault-free (or faulty) if the diagnosed node is fault-free (or faulty). However, the test performed by a faulty node is unreliable. Given a set of the

outcome of the tests (a.k.a "syndrome"), modeled as a graph whose vertices are nodes and edges are the test outcomes, the system can identify some or all faulty nodes. There has been plethora of work in system-level fault diagnosis [62, 63, 64, 65, 61, 66, 67]. Interested readers can start exploring this topic from an excellent survey in [68][69].

PMC mode and its variations are known as *invalidation* models. A second approach, known as the *comparison model*, has been introduced independently by Malek [70] (a.k.a *asymmetric comparison model*) and by Hakimi and Chwa [71] (a.k.a *symmetric comparison model*). In comparison models, a pair of nodes will be assigned to execute an identical test job. The agreements and disagreements among the nodes are the basis for identifying the set of faulty nodes. In both models, it is assumed that two fault-free nodes give matching test results while a faulty and a fault-free node give the mismatching results. The difference in the two models is the test outcome of two faulty nodes. In the symmetric model, both test outcomes are possible (i.e., matching and mis-matching outcome), while in the asymmetric model two faulty nodes always give the mis-matching test output. The comparison models are then extended to into Meang/Malek (MM) model [72], which later on was generalized into "generalized comparison model" by Sengupta and Dahbura [73]. Blough and Brown proposed the "broadcast comparison model" [74], in which two tested nodes broadcast their results to all other nodes. Chessa and Santi further applied the comparison-based model system-level diagnosis approach to ad-hoc networks [75].

Alibi, in some senses, shares several common characteristics with system-level fault diagnosis. They both consider the problem of identifying faulty/misbehaving nodes with the assumption that fault-free/honest nodes are willing to cooperate. They also specify the necessary condition to identify faulty/mis-behaving nodes where the system can identify all faulty/misbehaving nodes (i.e., one-step diagnosis) or one-by-one (i.e., sequential diagnosis). The comparison model has similar approach with receiving-based alibis. They both have a comparator test a pair of nodes and identify faulty/comprimised nodes based on the outcome of the test outcomes. However, alibi and system-level fault diagnosis have several fundamental differences.

- The fault model is different. In system-level fault diagnosis, the fault

can be hard or soft. In alibi, compromised nodes adversely avoid getting identified by exploiting the knowledge of the protocols used in the system.

- In system-level diagnosis, one important assumption is that a fault-free node can correctly and reliably test the faulty node. In the current design of alibis, this assumption does not hold.

- In broadcast comparison model whose model is closest to receiving-based alibi only considers asymmetric comparison model. In alibi, the asymmetric comparison model does not hold because attackers can share alibis and thus yield the same test result.

CHAPTER 8

CONCLUDING REMARKS AND FUTURE DIRECTIONS

8.1 Concluding Remarks

Dealing with insider attackers is always challenging because the attackers have the inside knowledge of the system. Most of the approaches to identifying insider attackers is to collect "malicious" behaviors of the nodes. However, such approach does not work if the attackers can manage to perform the attacks without revealing their identities. The insider jamming attacks presented in this dissertation is an example. A jammer can use reactive jamming attacks to corrupt the packets of other nodes in the network without revealing its identity.

To identify insider jammers, we propose a novel approach in which the system collects alibis of nodes to infer compromised nodes that perform reactive jamming attacks. The approach is complementary to the existing approaches because the system collects "good" behaviors of nodes instead of "malicious" behaviors.

We conclude the main contributions of this dissertation as follows.

- We present novel alibi frameworks for identifying insider jammers in single-hop wireless networks. The basic idea is to exploit the half-duplex nature of the jammers in which they cannot do both send and receive at the same time. Thus, if a node can present a proof showing that it is sending or receiving a packet while there is a jamming activity at the same time, it can prove that it cannot be the jammer at that time (i.e., it has an alibi). We expand the alibi framework into sending-based alibis and receiving-based alibis. The sending-based alibis rely on the fact that the jammers cannot send in two channels simultaneously while receiving-based alibis rely on the fact that the jammers cannot receive when they are sending. We also present alibi algorithms, protocols and

jamming-resistant communication that can be used in any instance of the alibi framework.

- We present three specific designs of the alibi framework including algorithms, protocols and architectures for three different types of networks: multi-channel WLANs, single-channel WLANs and multi-channel ad-hoc networks. The designs specifically show how S-alibis and R-alibis are used to facilitate the identification process in different networks. Specifically, for multi-channel WLANs, we propose the usage a hybrid of sending-based alibis and receiving-based alibis (Chapter 4). For single-channel WLANs and multi-channel ad hoc networks, we propose the usage of receiving-based alibis (Chapter 5 and 6). The analytical results, simulation results and experimental results confirm the hypothesis that a jammer can be identified when its jamming probability exceeds certain threshold. The threshold is a function of the sending/receiving rates of other nodes and and the jamming rates of other attackers.

8.2 Limitations

The current designs of the alibi framework have several limitations. First, the alibi framework may have significant communication overhead for exchanging alibis in the network. To reduce the communication overhead, we proposed that each alibi framework should have 1) a reactive defense mechanism in which the proofs and alibis are exchanged only when the jamming attacks are detected and 2) a mechanism to compress proofs and alibis.

Second, the sending-based alibi framework presented in Chapter 4 has a strong assumption: it requires all honest nodes to have non-zero sending rate. If a node does not have anything to send, it will be accused as an attacker. This limitation is due to the nature of sending-based alibis: alibis are granted based on the sending actions of nodes.

Third and last, the design of receiving-based alibi framework presented in Chapter 6 cannot cope with colluding attackers that share the corrupted packets among themselves. This limitation is due to the property of receiving-based alibis: corrupted proofs cannot be authenticated and thus can be

shared arbitrarily.

8.3 Future Directions

There are several future directions based on this work.

- *Defending against colluding attacks and Sybil attacks*: Receiving-based alibis still have some problems with the colluding and Sybil attacks. Defending against these attacks will make receiving-based alibis much stronger.

- *Decentralized detection.* The current alibi framework relies on a trusted central detector. One major goal will be to decentralize the detection process.

- *Multi-hop networks.* The dissertation mainly considers the single-hop wireless networks. Extending the alibi framework to multi-hop wireless networks is important. However, jamming defense in multi-hop wireless networks will be challenging. First, a jammer may have different jamming area in the network which makes the identification more complicated. Second, the jammers may block any particular routing process to prevent the alibi-exchange among nodes. The design of the alibi framework will need to take these factors into account.

- *Large-scale test-bed.* The dissertation has experiments on small scale test-bed. Alibi framework evaluation on large-scale networks is done through simulation. It is very interesting to see the alibi framework on a large-scale test-bed.

- *System-level fault diagnosis and alibis*: There might be connections between system-level fault diagnosis and alibis. If alibi is defined properly, a lot of results in system-level fault diagnosis can be applicable to alibis.

REFERENCES

[1] L. Fried, "Wave bubble: A design for a self-tuning portable RF jammer, http://www.ladyada.net/make/wavebubble/."

[2] D. J. Thuente and M. Acharya, "Intelligent jamming in wireless networks with applications to 802.11b and other networks," in *Military Communications Conference (MILCOM)*, 2006.

[3] W. Xu, W. Trappe, Y. Zhang, and T. Wood, "The feasibility of launching and detecting jamming attacks in wireless networks," in *ACM International symposium on Mobile ad hoc networking and computing (MobiHoc)*, 2005, pp. 46–57.

[4] W. Xu, T. Wood, W. Trappe, and Y. Zhang, "Channel surfing and spatial retreats: defenses against wireless denial of service," in *Proceedings of the 3rd ACM workshop on Wireless security (WiSe)*, 2004, pp. 80–89.

[5] J. T. Chiang and Y.-C. Hu, "Dynamic jamming mitigation for wireless broadcast networks," in *IEEE Conference on Computer Communications (INFOCOM)*, 2008.

[6] W. Xu, W. Trappe, and Y. Zhang, "Channel surfing: defending wireless sensor networks from interference," in *International conference on Information processing in sensor networks (IPSN)*, 2007, pp. 499–508.

[7] R. Negi and A. Perrig, "Jamming analysis of MAC protocols," Carnegie Mellon Technical Memo, Tech. Rep., 2003.

[8] "IEEE standard 802.11," September 2004.

[9] V. Navda, A. Bohra, S. Ganguly, and D. Rubenstein, "Using channel hopping to increase 802.11 resilience to jamming attacks," in *IEEE Conference on Computer Communications (INFOCOM) Minisymposium*, Anchorage, AK, May 2007.

[10] S. Khattab, D. Mosse, and R. Melhem, "Jamming mitigation in multiradio wireless networks: Reactive or proactive?" in *International conference on Security and privacy in communication netowrks (SecureComm)*, 2008.

[11] X. Liu, R. Rajaraman, G. Noubir, B. Thapa, C. King, and E. Bayraktaroglu, "On the performance of IEEE 802.11 under jamming," in *IEEE Conference on Computer Communications (INFOCOM)*, 2008.

[12] A. Chan, X. Liu, G. Noubir, and B. Thapa, "Mitigating control-channel jamming attacks in multi-channel ad hoc networks," in *IEEE International Symposium on Information Theory (ISIT)*, 2007.

[13] L. Lazos, S. Liu, and M. Krunz, "Mitigating control-channel jamming attacks in multi-channel ad hoc networks," in *ACM conference on Wireless network security (WiSec)*. ACM, 2009, pp. 169–180.

[14] M. Strasser, C. Pöpper, and S. Čapkun, "Efficient uncoordinated fhss anti-jamming communication," in *ACM International symposium on Mobile ad hoc networking and computing (Mobihoc)*. ACM, 2009, pp. 207–218.

[15] M. Strasser, C. Pöpper, S. Capkun, and M. Cagalj, "Jamming-resistant key establishment using uncoordinated frequency hopping," in *IEEE Symposium on Security and Privacy (Oakland)*. IEEE Computer Society, 2008, pp. 64–78.

[16] T. Jin, G. Noubir, and B. Thapa, "Zero pre-shared secret key establishment in the presence of jammers," in *ACM International Symposium on Mobile ad hoc networking and computing (Mobihoc)*, 2009.

[17] C. Popper, M. Strasser, and S. Capkun, "Jamming-resistant broadcast communication without shared key," in *USENIX security Symposiump*, 2009.

[18] Y. Liu, P. Ning, H. Dai, and A. Liu, "Randomized differential dsss: Jamming-resistant wireless broadcast communication," in *IEEE Conference on Computer Communications (INFOCOM)*, 2010.

[19] L. C. B. III, W. L. Bahn, , and M. D. Collins, "Jam-resistant communication without shared secrets through the use of concurrent codes," U.S. Air Force Academy, Tech. Rep., 2007.

[20] A. Wald, "Sequential analysis," in *J. Wiley & Sons*, 1947.

[21] Z. Golebiewski, M. Klonowski, M. Koza, and M. Kutylowski, "Towards fair leader election in wireless networks," in *International Conference on Ad-Hoc, Mobile and Wireless Networks (AdHoc-NOW)*. Springer-Verlag, 2009, pp. 166–179.

[22] P. Bahl, R. Chandra, and J. Dunagan, "SSCH: slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks," in *International Conference on Mobile Computing and Networking (MOBICOM)*, 2004, pp. 216–230.

[23] H.-S. W. So and J. Walrand, "McMAC: A multi-channel MAC proposal for ad-hoc wireless networks," UCB Tech. Report, Tech. Rep., 2005.

[24] J. Mo, H.-S. W. So, and J. Walrand, "Comparison of multi-channel MAC protocols," in *ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM)*, 2009.

[25] A. Perrig, R. Canetti, J. D. Tygar, and D. Song, "The TESLA broadcast authentication protocol," in *RSA CryptoBytes*, 2002.

[26] L. Sang and A. Arora, "Capabilities of low-power wireless jammers," in *IEEE Conference on Computer Communications (INFOCOM)*, 2009.

[27] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing (STOC)*, 2002.

[28] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," in *ACM symposium on Theory of computing (STOC)*, 1998.

[29] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher, "Min-wise independent permutations (extended abstract)," in *ACM symposium on Theory of computing (STOC)*, 1998.

[30] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proceedings of the twentieth annual symposium on Computational geometry (SCG)*, 2004.

[31] G. S. Manku, A. Jain, and A. D. Sarma, "Detecting near-duplicates for web crawling," in *International conference on World Wide Web (WWW)*, 2007.

[32] B. Awerbuch, A. Richa, and C. Scheideler, "A jamming-resistant MAC protocol for single-hop wireless networks," in *ACM symposium on Principles of distributed computing (PODC)*, 2008.

[33] A. D. Wood, J. A. Stankovic, and G. Zhou, "DEEJAM: Defeating energy-efficient jamming in IEEE 802.15.4-based wireless networks," in *IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, 2007.

[34] I. Shin, Y. Shen, Y. Xuan, M. T. Thai, and T. Znati, "Reactive jamming attacks in multi-radio wireless sensor networks: an efficient mitigating measure by identifying trigger nodes," in *ACM International workshop on Foundations of wireless ad hoc and sensor networking and computing (FOWANC)*. ACM, 2009, pp. 87–96.

[35] E. Felstead, "Follower jammer considerations for frequency hopped spread spectrum," in *IEEE Military Communications Conference (MILCOM)*, vol. 2, 1998, pp. 474–478.

[36] Z. Zhang, J. Wu, J. Deng, and M. Qiu, "Jamming ack attack to wireless networks and a mitigation approach," in *IEEE Global Telecommunications Conference (GLOBECOM)*, 2008, pp. 4966–4970.

[37] S. Khattab, D. Mosse, and R. Melhem, "Modeling of the channel-hopping anti-jamming defense in multi-radio wireless networks," in *International Conference on Mobile and Ubiquitous Systems (Mobiquitous)*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008, pp. 1–10.

[38] G. Alnifie and R. Simon, "A multi-channel defense against jamming attacks in wireless sensor networks," in *Proceedings of the 3rd ACM workshop on QoS and security for wireless and mobile networks (Q2SWinet)*, 2007, pp. 95–104.

[39] J. M. McCune, E. Shi, A. Perrig, and M. K. Reiter, "Detection of denial-of-message attacks on sensor network broadcasts," in *IEEE Symposium on Security and Privacy*, 2005, pp. 64–78.

[40] Y. W. Law, L. van Hoesel, J. Doumen, P. Hartel, and P. Havinga, "Energy-efficient link-layer jamming attacks against wireless sensor network MAC protocols," in *ACM workshop on Security of ad hoc and sensor networks (SASN)*, 2005.

[41] R. K. Mallik, R. A. Scholtz, and G. P. Papavassilopoulos, "Analysis of an on-off jamming situation as a dynamic game," *IEEE Transaction on Communications*, vol. 48, pp. 1360–1373, 2000.

[42] M. Li, I. Koutsopoulos, and R. Poovendran, "Optimal jamming attacks and network defense policies in wireless sensor networks," in *IEEE Conference on Computer Communications (INFOCOM)*, 2007.

[43] T. X. Brown, J. E. James, and A. Sethi, "Jamming and sensing of encrypted wireless ad hoc networks," in *ACM International symposium on Mobile ad hoc networking and computing (MobiHoc)*, 2006, pp. 120–130.

[44] P. Kyasanur and N. H. Vaidya, "Detection and handling of MAC layer misbehavior in wireless networks," in *International Conference on Dependable Systems and Networks (DSN)*, vol. 00, 2003, p. 173.

[45] P. Tague, M. Li, and R. Poovendran, "Probabilistic mitigation of control channel jamming via random key distribution," in *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2007.

[46] P. Tague, M. Li, and R. Poovendran, "Mitigation of control channel jamming under node capture attacks," *IEEE Transactions on Mobile Computing*, vol. 8, no. 9, pp. 1221–1234, 2009.

[47] A. D. Wood, J. A. Stankovic, and S. H. Son, "JAM: A jammed-area mapping service for sensor networks," in *IEEE Real-Time Systems Symposium (RTSS)*, 2003.

[48] O. Sidek and A. Yahya, "Reed solomon coding for frequency hopping spread spectrum in jamming environment," *American Journal of Applied Sciences*, pp. 1281–1284, 2008.

[49] W. Chen, D. Chen, G. Sun, and Y. Zhang, "Defending against jamming attacks in wireless local area networks," in *Autonomic and Trusted Computing*, 2007, pp. 519–528.

[50] M. Z. Win and R. A. Scholtz, "Ultra-wide bandwidth time-hopping spread-spectrum impulse radio for wireless multiple-access communications," *IEEE Transactions on Communications*, vol. 48, 2000.

[51] E. Rescorla, "Diffie-hellman key agreement method," RFC 2631.

[52] S. Radosavac, J. S. Baras, and I. Koutsopoulos, "A framework for MAC protocol misbehavior detection in wireless networks," in *ACM workshop on Wireless security (WiSe)*. New York, NY, USA: ACM, 2005, pp. 33–42.

[53] S. Radosavac, G. Moustakides, J. Baras, and I. Koutsopoulos, "An analytic framework for modeling and detecting access layer misbehavior in wireless networks," *ACM Transactions on Information and System Security*, vol. 11, no. 4, pp. 1–28, 2008.

[54] M. Raya, J.-P. Hubaux, and I. Aad, "DOMINO: a system to detect greedy behavior in IEEE 802.11 hotspots," in *International conference on Mobile systems, applications, and services (Mobisys)*. ACM, 2004, pp. 84–97.

[55] Y. Rong, S. K. Lee, and H.-A. Choi, "Detecting stations cheating on backoff rules in 802.11 networks using sequential analysis," in *IEEE Conference on Computer Communications (INFOCOM)*, 2006.

[56] K. Pelechrinis, G. Yan, S. Eidenbenz, and S. V. Krishnamurthy, "Detecting selfish exploitation of carrier sensing in 802.11 networks," in *IEEE Conference on Computer Communications (INFOCOM)*, 2009, pp. 657–665.

[57] Q. Zhang, T. Yu, and P. Ning, "A framework for identifying compromised nodes in wireless sensor networks," *ACM Transactions on Information and System Security (TISSEC)*, vol. 11, no. 3, pp. 1–37, 2008.

[58] M. Cagalj, S. Ganeriwal, and J.-P. Hubaux, "On selfish behavior in CSMA/CA networks," in *IEEE Conference on Computer Communications (INFOCOM)*, 2005.

[59] L. Chen and J. Leneutre, "Selfishness, not always a nightmare: Modeling selfish MAC behaviors in wireless mobile ad hoc networks," in *International Conference on Distributed Computing Systems (ICDCS)*. IEEE Computer Society, 2007, p. 16.

[60] J. Konorski, "A game-theoretic study of CSMA/CA under a backoff attack," *IEEE/ACM Transaction on Networking*, vol. 14, no. 6, pp. 1167–1178, 2006.

[61] F. P. Preparata, G. Metze, and R. T. Chien, "On the connection assignment problem of diagnosable systems," *IEEE Transactions on Electronic Computers*, vol. EC-16, no. 6, pp. 848 –854, dec. 1967.

[62] N. H. Vaidya and D. K. Pradham, "Safe system level diagnosis," *IEEE Transaction on Computers*, vol. 43, no. 3, pp. 367–370, 1994.

[63] N. Vaidya and D. Pradhan, "System level diagnosis: Combining detection and location," in *Fault-Tolerant Computing, 1991*, 25-27 1991, pp. 488 –495.

[64] F. Barsi, F. Grandoni, and P. Maestrini, "A theory of diagnosability of digital systems," *IEEE Transactions on Computers*, vol. 25, no. 6, pp. 585–593, 1976.

[65] S. E. Kreutzer and S. L. Hakimi, "System-level fault diagnosis: A survey," *Microprocessing and Microprogramming*, vol. 20, no. 4-5, pp. 323–330, 1987.

[66] J. Xu and S. ze Huang, "Sequentially t-diagnosable systems: A characterization and its applications," *IEEE Transaction on Computers*, vol. 44, no. 2, pp. 340–345, 1995.

[67] T. Araki and Y. Shibata, "(t, k)-diagnosable system: A generalization of the PMC models," *IEEE Transaction on Computers*, vol. 52, no. 7, pp. 971–975, 2003.

[68] M. Barborak, A. Dahbura, and M. Malek, "The consensus problem in fault-tolerant computing," *ACM Computing Survey*, vol. 25, no. 2, pp. 171–220, 1993.

[69] S. Lee and K. G. Shin, "Probabilistic diagnosis of multiprocessor systems," *ACM Computing Survey*, vol. 26, no. 1, pp. 121–139, 1994.

[70] M. Malek, "A comparison connection assignment for diagnosis of multiprocessor systems," in *Annual symposium on Computer Architecture (ISCA)*. ACM, 1980, pp. 31–36.

[71] K.-Y. Chwa and S. L. Hakimi, "Probabilistic diagnosis of multiprocessor systems," *Information and Control*, vol. 49, no. 3, pp. 212–238, 1981.

[72] J. Maeng and M. Malek, "A comparison connection assignment for self-diagnosis of multiprocessor systems," in *International Symposium Fault-Tolerant Computing*, 1981, pp. 173–175.

[73] A. Sengupta and A. T. Dahbura, "On self-diagnosable multiprocessor systems: Diagnosis by the comparison approach," *IEEE Transactions on Computers*, vol. 41, no. 11, pp. 1386–1396, 1992.

[74] D. M. Blough and H. W. Brown, "The broadcast comparison model for on-line fault diagnosis in multicomputer systems: Theory and implementation," *IEEE Transactions on Computers*, vol. 48, no. 5, pp. 470–493, 1999.

[75] S. Chessa and P. Santi, "Comparison-based system-level fault diagnosis in ad hoc networks," *IEEE Symposium on Reliable Distributed Systems*, pp. 257–266, 2001.