University of Nevada, Reno

Survivability Against Intelligent Adversary in Next-Generation Wireless Networks

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science and Engineering

by
Suman Bhunia

Shamik Sengupta / Dissertation Advisor

May 2017

© by Suman Bhunia 2017
All Rights Reserved

UNIVERSITY OF NEVADA RENO

THE GRADUATE SCHOOL

We recommend that the dissertation prepared
under our supervision by

SUMAN BHUNIA

entitled

Survivability Against Intelligent Adversary in Next-Generation Wireless Networks

be accepted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

Shamik Sengupta, Ph.D. – Advisor

Murat Yuksel, Ph.D. – Committee Member

Sergiu Dascalu, Ph.D. – Committee Member

Mehmet Hadi Gunes, Ph.D. – Committee Member

Sankar Mukhopadhyay, Ph.D. – Graduate School Representative

David Zeh, Ph.D. – Dean, Graduate School

May 2017

Abstract

The conventional static spectrum allocation policy has resulted in a suboptimal use of spectrum resources, leading to over-utilization in some bands and under-utilization in others. As a solution, dynamic spectrum access-based Cognitive Radio Network (CRN) has been proposed. CRN allows secondary users (SUs) to use an unused licensed spectrum while the proprietary primary user (PU) is not transmitting. CRN being a next generation wireless network inherits all the challenges of wireless and brings some critical issues due to the dynamic spectrum sensing and acquirement. Multiple SUs compete for spectrum and create conflicts and collisions in spectrum acquirement. An adversary can intelligently exploit these vulnerabilities to disrupt the communications of legitimate SUs. In this research, we address these unique challenges in the battle for coexistence.

We first present a framework for dynamic spectrum allocation with aggregation and fragmentation. Fitting the spectrum requirement of multiple SUs is an NP-hard problem. We propose three different techniques for spectrum allocation optimization: centralized, decentralized, and another hybrid solution with leader election. In this battle for coexistence, the broadcasting and open nature of transmission leaves a CRN open to jamming based Denial of Service (DoS) attacks. Since SUs use different channels for communication and attacker is also capable of attacking one channel, an intelligent attacker has to choose the DoS target by sensing all possible channels. We propose CR-Honeynet, a framework that exploits the intelligence of an attacker and lures it to a decoy transmission, while other legitimate communications bypass attacks. However, selecting a node to act as

decoy degrades its performance. We propose state-based decoy selection strategies that select decoys dynamically based on optimization criteria that deals with queue length, service type and arrival rate. and optimizes the overall end-to-end system performance. We then model the battle of defender and attacker from a game-theoretic point of view, where both parties are intelligent and learn about heterogeneous channel utilities from history. The theoretical model showes that there exists a Nash equilibrium for learning period of both players and if they deviate, the other party wins. We extend our study to attackers with the capability of moving in all three directions where adaptive beamnulling is useful to filter the signal coming from a jammer spatially. We propose a tracking based framework to optimize the beamnull that minimizes the risk of attack while minimizing link failure. Last but not that leastFinally, we develop a state-of-the-art testbed with off-the-shelf devices to evaluate the performance of the proposed framework.

Dedicated to my beloved parents.

Acknowledgements

I would like to thank my advisor Dr. Shamik Sengupta without whom I would not have finished my Ph.D. He was supportive throughout my research process, was patient beyond measures, and always ready to extend generous help whenever I needed it. I would also like to express my gratitude to the rest of my thesis committee: Dr. Murat Yuksel, Dr. Sergiu Dascalu, Dr. Mehmet Hadi Gunes, and Dr. Sankar Mukhopadhyay for their continuous encouragements, kind support, insightful comments for not only my thesis but also for the life of a researcher. I am grateful to Dr. Felisa Vázquez-Abad and Dr. Saad Mneimneh for their help and support during the collaboration work in my Ph.D. thesis. I am indebted to our department chair, Dr. Bebis, and the Dean of Graduate school, Dr. Zeh, whose continuous motivation and suggestions have helped me succeed as an academician. The members of the CSE department have been generous when they provided their constructive suggestions and critique, and I appreciate their help. I am grateful to all my colleagues in GSA whom I had a chance to work with, for their help, and critical comments in the professional life. I would like to thank all my colleagues at computer network laboratory for maintaining an excellent discussion friendly working environment and inspiring me for doing better research works. I would like to especially thank my collaborators Mahmudur, Paulo, Deepak, and Vahid for their support and help in my thesis work. I would like to mention the inspiration that I received from the volunteers of Akhil Bharat Vivekananda Yuva Mahamandal. Their works in relief work in rural disaster motivated me to research on an intelligent wireless network that can survive in adversarial condition. Last, but definitely not the least, I would like to thank my parents for their selfless support throughout my life. Without their emotional and moral support, I would not have reached so far.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the past two decades, wireless data communication demand has increased exponentially thanks to the advancement in the technology. Data carrying capacity of a transceiver is limited by the allocated spectrum. Regulatory bodies such as Federal Communication Commission (FCC) allocate spectrum to different organizations in a static manner [9]. In the conventional static spectrum allocation policy, a huge portion of the radio frequency is allocated to the government, TV broadcasting, military, and public safety systems where spectrum usages are significantly low. This has resulted in a sub-optimal use of spectrum resources, leading to over-utilization in some bands and under-utilization in others [10]. Fig. 1.1 shows a spectrum occupancy chart where we can easily identify the underutilized spectrum bands.

As a solution, dynamic spectrum access (DSA) based spectrum management policies are proposed [9, 11]. DSA allows a secondary user (SU) to opportunistically access the spectrum, either left unused by a proprietary primary user (PU), i.e., the owners of spectrum rights, or unlicensed spectrum that must be shared among different networks. Conventional radios can only access one area of the radio spectrum, which is selected at the time of design. Because of this, they do not meet the

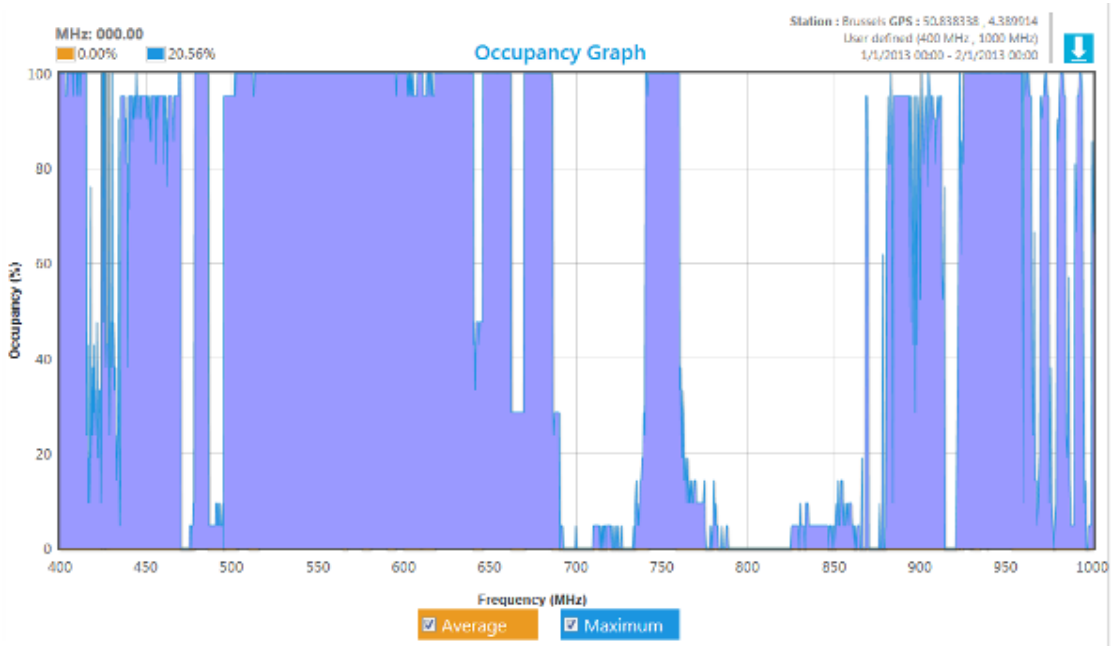


FIGURE 1.1: Spectrum occupancy map (Courtesy: Microsoft) [1]

adaptability requirements of DSA scenarios. The emergence of Software Defined Radios (SDRs) is the first step in making DSA networks feasible. An SDR can be defined as a radio where transmission frequencies, modulation type, and other Radio Frequency (RF) parameters can be configured and reconfigured by software. As a consequence, they can provide a broad range of services with variable Quality of Service (QoS) to adapt to different network technologies and the dynamics of radio propagation.

The addition of cognition capability to SDRs leads to the idea of a Cognitive Radio (CR), an intelligent radio capable of tuning itself based on its perception of the spectrum availability and the environment conditions. Recent improvements in computational abilities of current electronic devices and in artificial intelligence have led to the consideration of CR technology as a feasible solution to the overcrowding of the spectrum space. The IEEE 802.22 [12], which is an emerging standard for CR-based wireless regional area networks (WRANs), aims at a vacant licensed TV spectrum to be used by SU without causing interference to PU. There are two main characteristics of CR: 1)cognitive capability, which refers to the

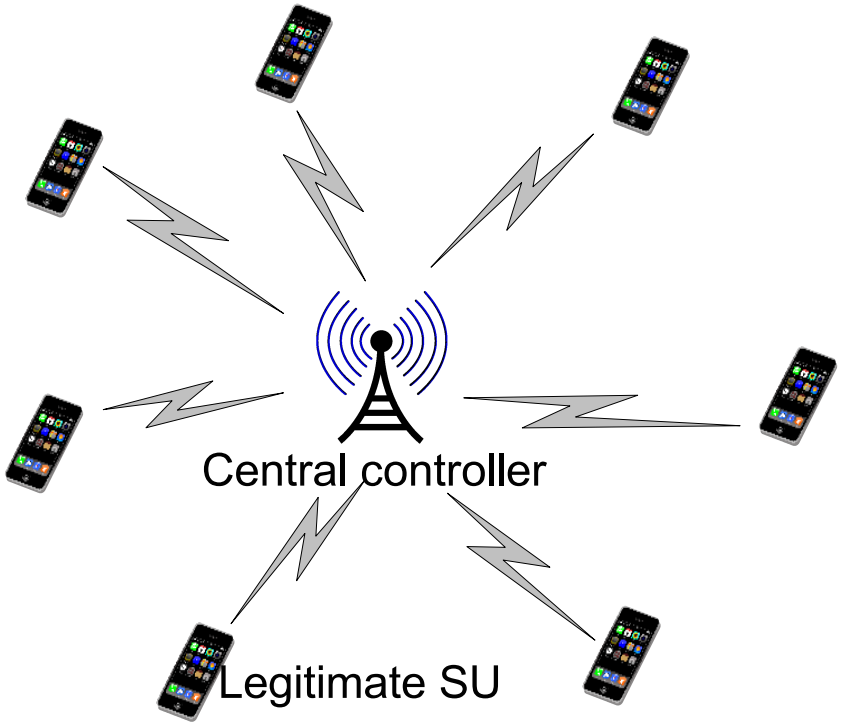


FIGURE 1.2: A sample CRN

ability of the CR to capture information from its radio environment and identify the best spectrum portion and operating parameters such as modulation, coding, power transmission, etc. 2) reconfigurability, that enables the radio components to be dynamically programmed according to cognitive decisions. Infrastructure-based cognitive radio networks (CRNs) consist of two major components: a central controller (such as base station or access point) and mobile SUs. The central controller supervises the communication and makes the spectrum allocation decisions. A sample CRN is presented in Fig. 1.2.

A CR node should be capable of frequency agility, dynamic frequency selection, location awareness, adaptive modulation, power control, and adapt transmission parameters dynamically [13]. It senses for spectrum holes and can adapt itself to transmit on spectrum opportunities found through spectrum sensing. Even though the PU protection mechanisms have been proactively studied, neither the secondary-secondary interaction mechanisms nor the protection of secondary users from malicious disruption has been specifically defined or addressed [10, 14, 15].

The fight for spectrum acquirement in CRN poses several challenges that are unique to CRN only. In the subsequent sections, we list the challenges.

1.1 Spectrum fragmentation and guard bands

DSA allows unlicensed or Secondary Users (SUs) to sense the wireless environment and opportunistically utilize idle portions of the spectrum known as holes, to establish their communication links. As the number of users and their requirements vary over time, the process of dynamic allocation creates spectrum holes which are too narrow to satisfy bandwidth requirements of a SU. In such cases, multiple narrow holes can be aggregated to achieve the minimum requirements. Several ideas for dynamic channel aggregation in DSA have been proposed. One approach enables aggregation by using multiple radio interfaces in CRs, where each radio interface can access only one contiguous spectrum hole [16]. The obvious constraint of this solution is that it limits the maximum number of accessible holes to the number of available radio interfaces. A popular alternative is Non-Contiguous Frequency Division Multiplexing (NC-OFDM), in which non-contiguous subsets of OFDM subcarriers are assigned to a single radio to establish an aggregated channel. This approach has shown an improvement in spectrum utilization and the overall performance of DSA networks [17–19]. However, it comes with an inherent overhead; every fragment of an aggregated channel requires guard bands on both sides to suppress cross-channel interference. These guard bands cannot be utilized for data transmission and therefore degrade spectrum utilization.

The severity of this issue is illustrated in Figure 1.3, in which the spectrum allocation of different secondary and Primary Users (PUs) are shown over a period of time. At t0, 3 SUs represented by colored blocks of red, green and blue are all utilizing contiguous blocks of spectrum, with yellow regions depicting guard bands on boundaries of these blocks. Then, beginning at t1, the arrival of new PUs (black)

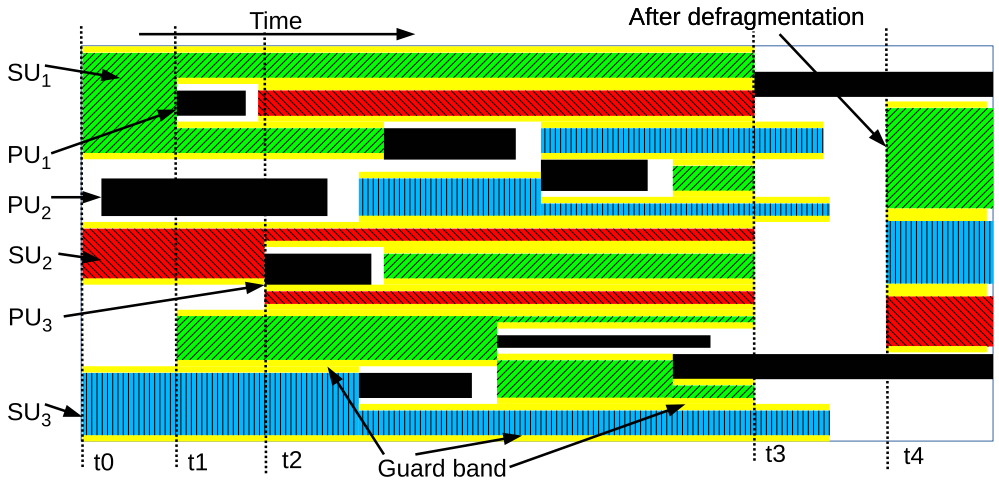


FIGURE 1.3: An example of spectrum fragmentation problem

forces the SUs to aggregate non-contiguous fragments and make up for the lost channels, hence more spectrum is dedicated to guard bands. At time t3, arrival of a PU leads to a situation where the total available spectrum is not sufficient for the SU depicted by red to adjust for its lost channels. Finally, time t4 shows that by rearranging non-contiguous fragments in bonded blocks, the bandwidth requirement of every user is met, as a smaller amount of spectrum is wasted on guard bands. It can be seen that increasing the number of fragments leads to a proportional increase in the number of guard bands, which are effectively wastage of spectrum as they cannot be utilized for data communications. Considering the purpose of DSA schemes is to achieve the highest spectrum utilization possible, mitigating techniques for the wastage issue must be investigated. In Chapter 3, we address this issue through periodic spectrum defragmentation. We consider three scenarios: a centralized method where a central controller takes care of the spectrum assignment, a decentralized one, and a hybrid solution where a group of nodes can elect a leader for spectrum defragmentation.

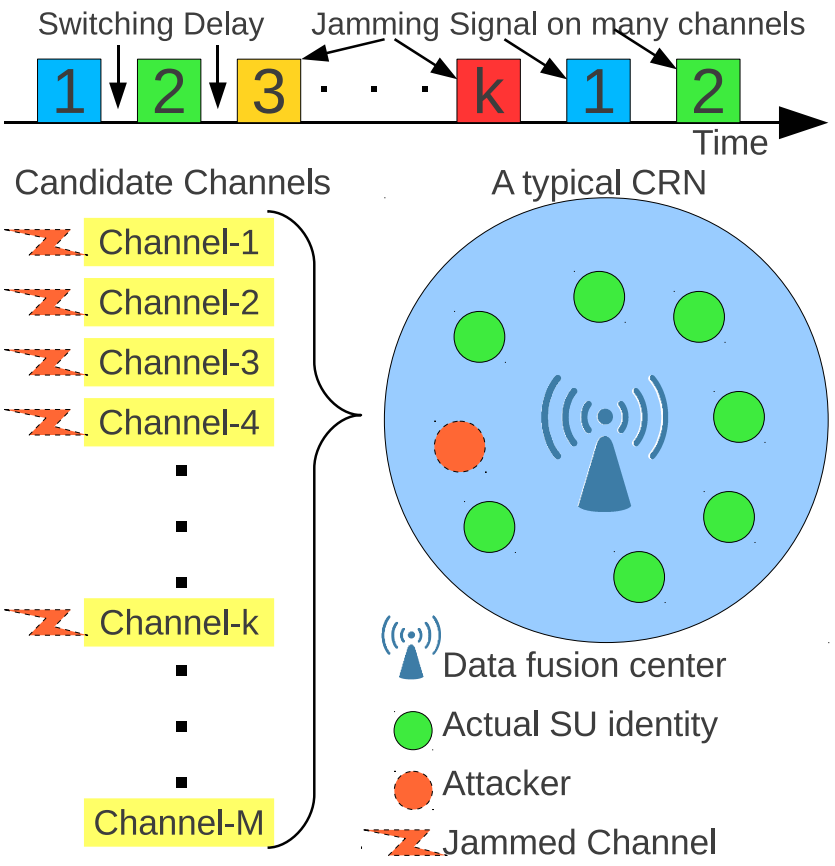


FIGURE 1.4: Schematic diagram for jamming based DoS attack

1.2 Jamming based denial of service attack

The "open" philosophy of the cognitive radio paradigm makes CRN susceptible to Jamming based Denial of Service (DoS) attacks by smart malicious users. An attacker can scan through channels, identify legitimate SU communications and then transmit a jamming signal on the same channel or fragment of the channel causing disruptive interference to the SU, which in effect can completely block the legitimate SU's transmission [10, 14, 15]. Fig. 1.4 presents a schematic of this kind of attack. Here the attacker is switching between several channels and transmitting jamming signals on several channels in TDMA manner to block several SU communications.

However, note that from an intelligent and rational attacker's perspective, jamming a communication randomly will not yield an optimal result; rather an attacker can be most disruptive if it targets the communication that impacts the CRN most severely upon interruption [15, 20–22]. The attacker succeeds in determining the highest impacting communication by observing certain transmission characteristics as highest transmission power, highest data rate, modulation scheme, packet inter-arrival time and quality of route with end-to-end acknowledgments [21, 23]. Again, an attacker may also use a combination of transmission characteristics instead of a single characteristic to discover the highest impacting communication. To defend against such attackers, a CRN must learn about the strategy (the targeted transmission characteristic(s)) that the attacker uses to discover the highest impacting communication. The attacker's strategy to find the highest impacting communication can be used as a trap by the defending CRN to detract the attacker from attacking legitimate communications.

1.3 Induced attack

After considering homogeneous channels, we focus on another vulnerability in CRN with heterogeneous channels, known as *induced attack*. A CRN learns the channel state and utility from its experience of transmission on that channel. In a heterogeneous multichannel DSA environment, a naive attacker searches for ongoing CRN transmission and transmits a jamming signal. An intelligent attacker not only disrupts the ongoing communication but also hampers CRN's long-term utility by inducing the CRN to use higher utility channels with lower probabilities. An attacker can induce the CRN to observe wrong channel information by intelligently interrupting the cognitive radios on a different channel. Successful induced attacks on candidate channel(s) will not only force the CRN to leave the channels but may also affect the estimation of channel utilities, because of the

"seemingly" low payoff from those channels. As a result, the learning mechanisms may discard certain channels as non-usable, even though in reality, the channels may be available or even have high payoffs. CRNs will be reluctant to consider these channels as potential candidate channels thus limiting their available radio resources.

In the presence of such adversarial scenarios, the problems of survivability, network management, and specifying performance bounds become highly challenging that must be addressed; else the performance of the secondary networks will be degraded defeating the purpose of DSA paradigm. The problem becomes more complex when we consider (1) channel utilities are no longer constant but can change abruptly due to PU's activities on the channel, changes in channel characteristics and interference/disruptions, and (2) channel utilities are unknown by the CRNs instantaneously, but they can be measured with noise whenever there is a successful transmission. Thus, a statistical estimation method must be used that incorporates regime change detection [24]. Under such dynamic regime changes, it becomes particularly complex to estimate and decide optimally in the presence of induced attacks. In Chapter 5, we develop a game between the CRN and adversary. We show that there exists an optimal learning period for both agents which will lead to their respective goals.

1.4 Autonomous movement of jammers in 3D

The ecosystem of wireless communications is evolving towards distributed, self-configuring ad hoc architectures. Elimination of the need for central communications infrastructure is beneficial in many scenarios as it allows seamless and quick deployment of agile networks. Such agility is an essential requirement for many applications, including emergency radio networks in disaster zones, tactical communications, and inter-vehicular networks. Also, commercialization of unmanned

aerial vehicles (UAVs) and the subsequent feasibility of multi-UAV missions introduce novel challenges and constraints on their network requirements, which may be adequately satisfied in the ad hoc manner. Following the same trend, the concept of 3D mesh networks is envisioned [25–28], in which aerial nodes collaborate with ground nodes to allow wider, more dynamic ad hoc deployments while enhancing spectrum utilization by exploiting spatial reuse.

Considering the advantages of ad hoc networking, it is envisioned that this paradigm will play a key role in future mission critical communications. Therefore, ensuring the security and robustness of such networks is essential for such applications. Even though the independence of ad hoc configurations from single points of failure is seen as a merit from the security point of view, their information flow is still susceptible to disruption by interference and jamming. Furthermore, it has been shown that jamming a subset of links in multihop networks is sufficient to incur maximal disruption of a network [29, 30]. Hence, mitigation of jamming attacks is a necessary component of mission-critical ad hoc networks.

Some well-known categories of anti-jamming techniques proposed in the literature are those that utilize specially designed signal coding and modulation, such as Frequency Hopping Spread Spectrum (FHSS) [31] and Direct Sequence Spread Spectrum (DSSS) [32]. The downside associated with this class of techniques is their larger bandwidth requirements. Considering the state of the overcrowded electromagnetic spectrum, this overload can prove to be costly. To preserve the scarce bandwidth, an alternative is to apply Spatial Filtering with beamforming antenna arrays [33]. This approach exploits the beamformers' ability to estimate the Direction of Arrival (DoA) of signals. This direction is then used to tailor the beamformer's response, such that the signals originating from sources of interference are suppressed or eliminated. Beamforming antenna systems that implement this mechanism are referred to as Adaptive Nulling Antennas (ANA) [34].

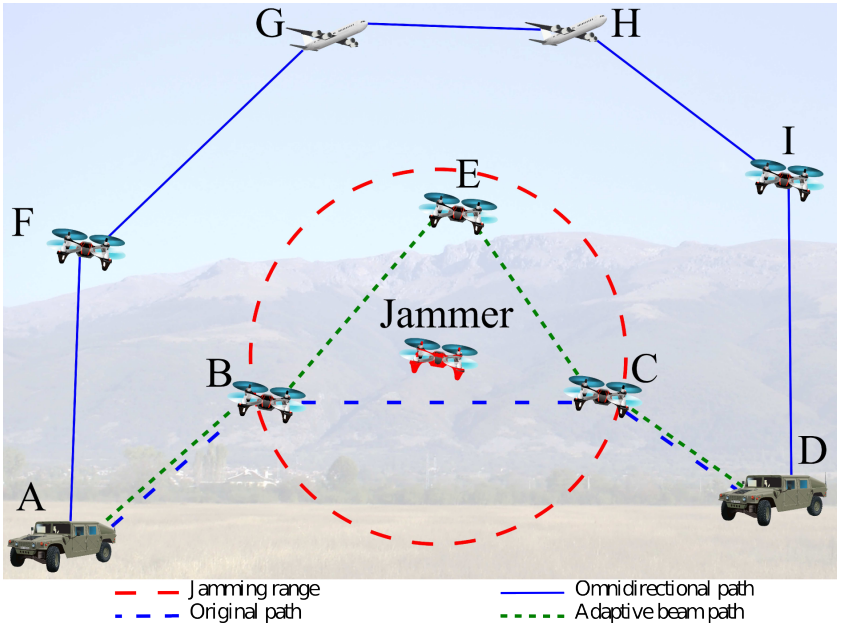


FIGURE 1.5: A comparison of routing in omnidirectional vs beam nulling schemes under jamming

Traditionally, ad hoc configurations assume that omnidirectional antennas are used for communications. In multihop networks, data is routed over multiple hops to reach a destination that is not within direct communication range of the source. By utilizing beam nulling techniques, a node can adapt its radiation pattern to create a null in the direction of interference. This allows for maintaining the links which are not affected by the jammer. Figure 1.5 provides an example of end-to-end data delivery in an ad hoc network. In the absence of a jammer, packets from A to D follow the path $A-B-C-D$ when all nodes employ omnidirectional antennas. In this configuration, the jammer can effectively neutralize nodes B, C and E. The routing protocol discovers the link failures and reroutes packets through $A-F-G-H-I-D$. This way packets are delivered at the cost of increased end-to-end delay, as well as congestion on link $G-H$.

However, when beam nulling is applied, nodes B, C and E can successfully avoid the jammer. Now packets can be delivered through $A-B-E-C-D$. Hence it can be seen that, in the presence of a jammer, adaptive beam nulling is not only capable of maintaining connectivity of the nodes inside the affected region, but

also ensures less congestion on the remaining links. The majority of the literature on ANAs rely on the assumption that the jammers are stationary with respect to beamformers [35–39]. However, with the recent expansion and growth of mobile wireless technologies, this assumption does not necessarily hold true. Also, there is a lack of publicly available analysis on the network performance of ad hoc networks utilizing adaptive nulling antennas under jamming.

We propose two distributed methods for beam nulling in multihop ad hoc networks. In these methods, the width and direction of null angles are calculated based on periodic sensing of the jammer's relative position to a node. As this measurement is not continuous, a prediction technique is introduced to estimate a jammer's movements until the next sensing phase. Measured and predicted locations are then incorporated in the calculation of a nulled region, which suppresses signals arriving from within its angular span. As signals coming from the neighboring nodes that fall within the null angle are also subject to suppression, the proposed method for calculation of null angle aims to minimize the number of legitimate link failures, while maximizing the confidence of jamming avoidance. The proposed methods also take randomness of the jammer's movements into account by introducing safety buffer zones on both edges of the null angle. To evaluate the effectiveness of this method for both 2D and 3D configurations, physical simulations are performed. The network-level performance of the proposed method is also evaluated by ns-3 simulations, where the impact of different ad hoc routing protocols on the overall performance of these methods are investigated.

1.5 Contribution and thesis organization

The rest of this thesis is organized as follows. In Chapter 2 we present the background study related to the defense mechanism against several attack scenarios. The Dynamic Spectrum Access testbed is described in Chapter 3. Chapter 4

presents the CR-Honeynode mechanism that proactively defends against jamming attacks in CRN. To build the mathematical model of the data transmission of CR-Honeynet, we first model a data transmission of CRN as queue with fixed server vacation where sensing interval is priority scheduled service. The concept of Induced attack is presented in Chapter 5. A stochastic game is presented which finds the optimal channel selection strategy for multiple heterogeneous channel environments in the presence of an induced attacker. Chapter 6 presents adaptive beamnulling to mitigate jamming in 3D UAV mesh networks. Finally, Chapter 7 concludes the dissertation.

Chapter 2

Background studies

In this section, we present a literature survey for three main topics: Spectrum allocation in DSA with aggregation and fragmentation, defense against jamming attack in cognitive radios, and adaptive beam nulling.

2.1 Spectrum allocation in DSA with fragmentation and aggregation

The problem of dynamic spectrum access has been explored through research for many years [40]. However, investigating channel aggregation and fragmentation has only recently gained attraction. While traditional spectrum allocation algorithms assign contiguous channels to users, wireless techniques such as NC-OFDM [41] provides the possibility of spectrum aggregation, in which multiple spectrum holes can be joined to satisfy the bandwidth requirements of a user [16][42]. Multiple Homogeneous and heterogeneous spectrum allocation schemes [43] have been investigated for DSA network. The spectrum aggregation and fragmentation can be used on top of them to reduce the spectrum wastage.

Aggregation Aware Spectrum Assignment (AASA) [17] is one of the earlier spectrum aggregation algorithms presented in the literature. This greedy algorithm is developed based on the assumption that all users require the same amount of spectrum and uses a first-fit approach for channel assignments. In this method, a broker searches for spectrum opportunities and assigns the available channels to users starting from the lowest frequency and moving upward. The authors showed that that AASA achieves a higher spectrum utilization than contiguous spectrum allocation schemes. In contrast to AASA, Maximum Satisfactory Algorithm (MSA) [18] is a best-fit algorithm developed for the case where users may have different spectrum requirements. In this approach, users with higher data-rate demand are prioritized as they are more difficult to fit in narrower spectrum holes. *Channel Characteristic Aware Spectrum Aggregation algorithm (CCASA)* [44] considers the heterogeneity of data carrying capacity in different parts of the spectrum. Once the channel state information of all users is known, a CCASA central controller allocates suitable spectrum fragments to the user by utilizing NC-OFDM. Using a sliding window method, CCASA calculates the maximum spectrum usage ratio for each user and allocates the spectrum to users in the decreasing order of their spectrum requirements. The work presented in [19] investigates fragmentation and aggregation in a software defined DSA prototype. They implemented a frequency agile testbed based on GNU Radio [45] and Ettus USRP [46] devices. Their framework includes a MAC overlay that senses the spectrum while receiving data, and once the requirements and access conditions change, the detected holes are dynamically allocated in a best-fit manner. If a single hole is not wide enough to satisfy a user's requirements, the user then aggregates multiple narrower holes using NC-OFDM.

Even with the considerable amount of theoretical and some practical investigations on channel aggregation, the issue of guard bands and their adverse effect on spectrum utilization and the overall network performance has not been studied in

the literature. We have addressed this issue in Chapter 3 by proposing algorithms for three different settings; centralized, distributed, and hybrid model, where the main contribution is the presentation of novel algorithms for solving this wastage.

2.2 Vulnerabilities in cognitive radios

Unlike conventional wired medium, a wireless network is vulnerable to many threats that may not be prevented by conventional security measures based on cryptography. CRN allows a secondary user to sense spectrum and dynamically transmit on different channels intelligently. This capability makes the wireless networks more vulnerable. Due to the broadcast nature of wireless medium, an adversary can monitor the messages and observe the transmissions characteristics. Attacking a proprietary user is not preferable for the attacker due to high penalties whereas attacking a secondary user (SU) is less risky, as the secondary user confuses the attack with that of a primary user arrival in the currently residing spectrum. Authors of [47] and [48] has broadly classified the motivation of attack into two categories. In *Selfish Attack*, secondary users in a network try to acquire more resources depriving others. In *malicious attack*, the adversary or rogue user tries to hamper the communication of other SUs, or it tries to steal information from legitimate communications.

Frankel et al. [49] has provided an overview of security threats in wireless networks. Cognitive Radio Networks are also vulnerable to these threats because the cognitive users communicate over shared wireless medium which is common to the intruders too.

- *Denial of Service* attack prevents authorized users from accessing resources.
- *Eavesdropping* attack monitors communication between authenticate users.

- *Man-in the-Middle* attack actively intercepts the path communication of genuine users. An attacker may obtain authentication credentials. In a wireless environment, a man-in-the-middle attack can be achieved by seizing control over the base station or access point.

- *Masquerading* attack entails attacker pretend as a legitimate and gain certain privileges.

- *Message Modification* can be done by an attacker by deleting or altering a message in the path of communication

- *Message Reply* is a special kind of attack where the attacker passively monitors recognized transmission and retransmits the message acting as if it is a valid user.

- *Traffic Analysis* can be done by an attacker, passively monitoring the legitimate communications and identify communication patterns by learning.

IEEE 802.22 based cognitive radio networks inherit all threats described above and are also vulnerable to many other issues due to their unique dynamic spectrum access characteristics. In CRN, an SU has to periodically sense the channel for primary user's presence. SU has to vacate the channel as soon as the primary user starts transmitting. These threats are described in [47, 50–60]. Fragkiadakis et al. [61] has broadly classified these threats to following categories:

- In *primary user emulation attack* [57], a malicious user mimics the transmission pattern of a primary user so that when other SUs sense the spectrum, they are deluded by detecting PU's transmission. In CRN, it causes false alarms about primary user transmission.

- In *spectrum sensing data falsification attack* [58], the rogue SU is part of a cooperative sensing group which can mislead the decision of the group

by providing wrong information intelligently. In [62], a utility function to determine bandwidth has been derived. A malicious user can report wrong information to gain higher bandwidth compared to others.

- The IEEE 802.22 standard enforces the overlapping CRN cell to be synchronized. A malicious agent can exploit this vulnerability and forge the inter-cell beacons. By tampering with the inter-cell beacons, the adversary reports false spectrum information to the neighboring cells. This kind of vulnerability is called *beacon falsification attack* [59].

- In *lion attack* [60], an adversary target to reduce the victim's throughput by forcing frequent channel switching. Channel switching entails latency in data communication as it involves sensing for the best channel to switch. In TCP connection, if the delay increases abruptly, TCP minimizes the contention window which causes a decrease in overall throughput.

- In *Parasite attack* [63], network links or channels with high priority are targeted as a victim. Attacker increases interference at heavily loaded links and thereby reducing system efficiency.

- In *Byzantine attack*, [64] a malicious agent modifies the spectrum sensing information for a particular goal. This modified or wrong information mislead central data fusion center to a wrong conclusion.

- *Sybil attack* [65] is particular type of attack where a malicious user can pose multiple forged identities. A malicious user can launch Byzantine attack in conjugation with Sybil attack which will make a great impact on the PU arrival event decision as according to majority rule it will override. This can severely degrade the performance of a network.

Among all the vulnerabilities specified above, PUE poses the greatest threat as it does not reveal the attackers' identity. A malicious entity can intentionally

cause harmful interference to a victim primary receiver as the receiver does not have location information of the transmitters. This is a Denial of Service (DoS) attack because the attacker emulates the primary user's transmission pattern and blocks all other impudent SUs to use that particular channel. Actual SUs in sensing period sense transmission of the attacker as a primary user presence and refrain from transmission in the next transmission period on that particular channel. The aim of a greedy user is to gain control of the channel by itself, on the other hand, a malicious attacker aims to disrupt the communication of others by inducing false alarm. In the cybersecurity arena, emulation attacks are prevalent, where malicious parties try to infiltrate target machines/organizations by emulating themselves as authenticated users. Cyber-threat information sharing [66–68] is considered as an alternative to understand the actions of adversaries proactively and possibly block them before they act.

Jamming can be broadly categorized into two types [69], [70]. In *physical layer jamming*, the attacker jams the channel of communication by sending strong noise or jamming signals. The *data-link / MAC layer jamming* targets several vulnerabilities present in the MAC layer protocol. Jamming essentially means disrupting communication of legitimate users.

Before acquiring spectrum, a CRN senses the spectrum and decides on the portion of the spectrum to use. It searches for opportunities in spectrum and then exploits the holes to communicate with other nodes. An SU senses a spectrum as blocked if it finds another transmission in that spectrum block. This blockage can be due to PUs and the competing SUs as well. Again malicious users intentionally transmit jamming signals emulating PUs so that the SUs don't use those channels.

Energy detection technique is mostly explored way to detect this kind of attack [61, 71]. learning mechanisms can be applied to detect attacks based upon the

history of energy on channels. Jin et al. Due to the noise in wireless medium, detection of jamming is crucial in combating an attacker. A good survey of different detection mechanisms for jamming based DoS attack has been presented in [21]. It is difficult to correctly detect jamming based on a single system parameter. Several system parameters, such as received-signal-strength, packet-send-ratio, packet-delivery-ratio, and carrier-sensing-time. are used for modeling jamming detection system. Consistency checks among system parameters are used for more efficient detection. Authors of [72] have classified spectrum usage anomaly detection data fusion algorithms. Through different fusion algorithms, anomalies in spectrum usage can be detected successfully with higher efficiency. A cross layer detection mechanism of anomalous spectrum attack has been proposed in [73], where the network maps the jammed geographical region using spectrum sensing reports sent from each SUs that are equipped with localization module.

2.3 Proposed defense mechanisms

In *Channel Surfing* technique, the node which is under attack migrates its channel of communication upon detection of jamming [70, 74]. Authors of [75] proposed proactive frequency hopping where the nodes change its channel of operation irrespective of attacks to avoid jamming. The authors considered fixed number of channel of the attacker that is known to an SU, which in reality is difficult to achieve. In *Spatial Retreat* [76] mobile nodes relocate themselves physically to avoid jamming. The constraint of this approach is that the nodes are required to be highly mobile which is not realistic for static nodes. In *Mapping Jammed Region* [77] approach, the multi-hop, and intensely populated CRN avoid routing through the links that have been affected by jamming. This mechanism fails if there is only one path and that link is attacked. [78] proposed a location-based analytical model for PUEA detection. Although actual location of a primary is

not required to be known by the secondary network. In this model they consider PU maintains a distance from the CRN, i.e. it is located outside the geographical border of a secondary network. The sensitivity of the detection can be controlled by providing a thresholds for the probability of missed detection and false alarms. The authors have assumed that malicious users can have constant transmission power which is not true for an intelligent attacker; hence, a further modification is needed. Again, location based scheme fails if the primary user is in the vicinity of secondary network or secondary network is mobile. Chen et al. [57] proposed a localization-based defense scheme which classifies the signal of the incumbent transmitter and of the attacker. Because this scheme requires the geographic model of the networks to be known a priori, it becomes infeasible for a high mobility scenario. A location based scheme is easy to implement as it requires energy level detection of the received signal.

In *Spread Spectrum* [79] technique, low-bandwidth data stream uses higher bandwidth channel to pass the information irrespective of jamming. Although this mechanism increases reliability of communication it provides very poor data rate. A single channel honeypot based channel surfing to mitigate jamming-based DoS attacks has been proposed in [69]. The network dedicates a node as a honeypot to monitor attacks and upon detection of an attack, the network switches its channel of operation which results in long time communication disruption. The majority of the previous works have assumed that the attacker is naive and does not evolve. Thus, none of these works have focused on learning the strategy of an attacker where the attacker is also dynamic and changes its strategy by choosing the target communication characteristics.

Mathur et al. [56] used a simplified digital signature key management to prevent an attacker from emulating a primary user. It employs a CA common to PU and secondary network. All SU transmissions are controlled by a base-station (BS) and all SUs communicate with BS through a licensed common control channel.

The primary user utilizes its digital signature to encrypt the data. When SU is suspicious about a possible attack, it stores the data received and send it to BS. BS can determine whether the signature is authentic with the help of CA. This mechanism doesn't require geographic information of the network and therefore can work in high mobility area. However it fails to operate when the primary user is an analog system such as TV signals.

Chapter 3

Fragmentation aware DSA

In the battle for spectrum acquirement, SUs fight with each other to find spectrum holes. If a hole can not meet the spectrum demand of an SU, it can aggregate multiple spectrum holes to construct a channel. The downside of this approach is that it can potentially waste vast amounts of spectrum as guard bands, and as the number of fragments increases the problem becomes more severe. This research work proposes online spectrum defragmentation as an effective solution to wastage of spectrum due to guard bands. In this method, changes in spectrum access, such as PUs vacating their channel or SUs changing their bandwidth requirements, defragmentation mechanism in the network are triggered that attempt to reduce the number of fragments by rearranging spectrum assignments thereby reducing the number of guard bands required. Three techniques are presented for different network architectures: 1) Infrastructure Networks with a Central Controller that runs the spectrum reallocation algorithm; 2) ad hoc network running a completely distributed defragmentation algorithm, and 3) ad hoc networks with a temporarily elected "leader" overseeing the defragmentation process. All three algorithms are theoretically studied and their effectiveness and efficiency are compared based on simulation results. Also, to study the practical aspects of this

approach, a prototype NC-OFDM DSA network is implemented with software defined radios, and the resultant hardware and network parameters are incorporated in the simulations.

The rest of this chapter is organized as follows: Section 3.1 formulates the problem and presents a detailed description and analysis of the defragmentation algorithms. Section 3.2 describes the prototype implementation and measurements, while Section 3.3 presents the simulation results and comparisons of the algorithms. Finally, Section 3.4 summarizes the chapter.

3.1 Theoretical analysis

Dynamic bandwidth requirements of SUs, as well as the varying nature of spectrum allocation in DSA networks, necessitate agile mechanisms for efficient utilization of every available space in spectrum. Even though in theory orthogonal carrier frequencies do not require frequency separation for suppression of adjacent-channel interference, it was shown in [19] that non-contiguous subsets of OFDM carriers have to be sufficiently separated to prevent destructive leakage of energy from one subset to another. This separation is referred to as guard band. It is intuitive that as the number of non-contiguous channels increases, more guard bands will be required. Also, OFDM transceivers use pilot subcarriers to perform coherent detection and reliable channel estimation [41]. These two constitute inherent overheads of a NC-OFDM system that negatively affect spectral utilization in DSA radios, which are fundamentally used to achieve the opposite by exploiting as much of the available spectrum as possible. Therefore, there is a fundamental need for a method of decreasing the overhead due to guard band allocation.

In this section, the underlying theory of one such method is presented, which is based on reduction of the number of fragments through online reassignment

and defragmentation of channels. Fundamental parameters considered by this method are not only the overhead caused by guard bands and pilot carriers, but also hardware limitations and the heterogeneity of the electromagnetic spectrum. Hence, this method takes into account the practical constraints of radio interfaces, and frequency selective fading of the environment to provide the most efficient solution. In the following, theoretical formulation of the spectrum assignment problem and the optimization goal are discussed. Then, the details of applying this framework in infrastructure and ad hoc DSA networks are presented.

3.1.1 Spectrum assignment problem

Let $\mathbb{N}$ be the set of all SUs where $\mathbb{N} = \{u_i | i = 1, \ldots, N\}$. The whole spectrum range is divided into C subcarriers. we define $\mathbb{C} = \{c_j | j = 1, \ldots, C\}$. Let T_i^{req} be the throughput demand of u_i, $i \in \mathbb{N}$. Let the throughput requirement matrix be $\mathcal{T}^{req} = (T_1^{req}, T_2^{req}, \cdots, T_N^{req})$. Due to heterogeneity in the wireless spectrum, different subcarriers provide different data rate [44, 80]. Let's define the data-rate matrix $\mathcal{R}$ which contains maximum data rate that can be achieved by u_i over subcarrier c_j is known.

$$\mathcal{R} = \begin{pmatrix} R_{1,1} & R_{1,2} & \cdots & R_{1,C} \\ \vdots & \vdots & \ddots & \vdots \\ R_{N,1} & R_{N,2} & \cdots & R_{N,C} \end{pmatrix}_{N \times C} \tag{3.1}$$

At any given time, PU usage matrix can be defined as

$$\mathcal{A}_{PU} = \begin{pmatrix} PU_1 & PU_2 & \cdots & PU_C \end{pmatrix}, A_j \in \{0, 1\}$$

Here $PU_j = 1$ if the subcarrier j is being used by its PU and 0 otherwise. Binary subcarrier assignment matrix is defined as:

$$\mathcal{A} = \begin{pmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,C} \\ \vdots & \vdots & \ddots & \vdots \\ A_{N,1} & A_{N,2} & \cdots & A_{N,C} \end{pmatrix}_{N \times C}, A_{i,j} \in \{0, 1\} \tag{3.2}$$

Where $A_{i,j} = 1$ if C_j is assigned to u_i, and 0 otherwise. We consider all SUs are within communication distance of each other. If multiple SUs transmit on the same subcarrier then the transmissions collide and data is lost. So, a subcarrier should not be assigned to an SU if another SU is using it or PU is occupying it. $\sum_{i \in \mathbb{N}} A_{i,j} + PU_j \leq 1, \forall j \in \mathbb{C}$.

In NC-OFDM transmission, the total allocated subcarriers consists of data, guard and pilot subcarriers. As the required guard band and pilot insertion are system and hardware dependent, we consider these two factors in the theoretical analysis. Let's define three $N \times C$ matrices: data-subcarrier assignment matrix ($\mathcal{D}$), pilot-subcarrier assignment matrix ($\mathcal{P}$) and guard-subcarrier assignment matrix ($\mathcal{G}$). Where $D_{i,j}, P_{i,j}, G_{i,j} \in \{0, 1\}$. $D_{i,j} = 1$ if channel j is allocated to u_i as data subcarrier at this point, and 0 otherwise. $P_{i,j}$ and $G_{i,j}$ follow same definition for pilot and guard subcarriers respectively. Essentially, $\mathcal{D} + \mathcal{P} + \mathcal{G} = \mathcal{A}$.

For u_i, the expected throughput is expressed as:

$$T_i = \begin{pmatrix} D_{i,1} & D_{i,2} & \cdots & D_{i,C} \end{pmatrix} \cdot \begin{pmatrix} R_{i,1} & R_{i,2} & \cdots & R_{i,C} \end{pmatrix}^T \tag{3.3}$$

Let's define cross interference matrix for subcarriers as:

$$\mathcal{I} = \begin{pmatrix} I_{1,1} & I_{1,2} & \cdots & I_{1,C} \\ \vdots & \vdots & \ddots & \vdots \\ I_{C,1} & I_{C,2} & \cdots & I_{C,C} \end{pmatrix}_{C \times C} \tag{3.4}$$

Where $I_{j,k}$ denotes the interference created by the transmission of j^{th} subcarrier on k^{th} subcarrier. To investigate a worst case scenario, nodes are assumed to be in close proximity of each other, and therefore, the effect of path loss on cross channel interference can be ignored. $I_{j,k} \in (0, 1)$ and $I_{j,j} = 1$. Value of $I_{j,k}$ for $j \neq k$ depends on the hardware. Subsequent guard bands must be placed to keep the interference under a threshold I_{th}. When an SU uses a set of subcarriers, it must ensure the transmission does not cause interference to other subcarriers. So, the constraint becomes,

$$\sum_{i \in \mathbb{N}} \sum_{k \in \mathbb{C} \wedge k \notin A_i} (D_{i,j} + P_{i,j}) I_{j,k} \leq I_{th}; \forall j \in \mathbb{C} \tag{3.5}$$

NC-OFDM transmission is limited by several hardware factors such as maximum aggregation limit (B_i), maximum usable subcarriers (C_i^{max}), maximum fragments per SU (F_i^{max}), maximum transmission power, etc [19, 44, 81]. If Cu_i and Cl_i are indices of highest and lowest allocated subcarrier for u_i then we can write, $B_i \leq Cu_i - Cl_i$. If u_i can only use C_i^{max} number of subcarriers as pilot or data subcarriers then, $\sum_{j \in \mathbb{C}} D_{i,j} + P_{i,j} \leq C_i^{max}$.

3.1.2 Optimization goal

For any spectrum allocation problem, the ultimate goal is to maximize the total achievable throughput $\sum_{i \in \mathbb{N}} T_i$ in the network. Equivalently, our goal translates

into maximizing the throughput while minimizing the number of allocated subcarriers to achieve that throughput. This can be written as:

$$
\begin{aligned}
\max \quad & \frac{\sum_{i\in\mathbb{N}} T_i}{\sum_{i\in\mathbb{N}}\sum_{j\in\mathbb{C}} A_{i,j}} \\
s.t. \quad & \sum_{i\in\mathbb{N}} \sum_{k\in\mathbb{C}\wedge k\notin A_i} (D_{i,j}+P_{i,j})I_{j,k} \leq I_{th}; \forall j \in \mathbb{C} \\
& T_i \geq T_i^{req}, \forall i \in \mathbb{N} \\
& PU_j + \sum_{i\in\mathbb{N}} A_{i,j} \leq 1, \forall j \in \mathbb{C} \\
& A_{i,j} \in \{0,1\}, \forall i \in \mathbb{N}, \forall j \in \mathbb{C} \\
& Cu_i - Cl_i \leq B_i; \forall i \in \mathbb{N} \\
& \sum_{j\in\mathbb{C}} D_{i,j} + P_{i,j} < C_i^{max}; \forall i \in \mathbb{N} \\
& \sum_{j\in\mathbb{C}} D_{i,j} + P_{i,j} \leq C_i^{max} \\
& \text{no. of fragments} \leq F_i^{max}
\end{aligned}
\tag{3.6}
$$

Theorem 3.1. *The throughput maximization problem in eq. 3.6 is NP-hard even if there is no PU present.*

Proof: The proof follows the reduction of the *0-1 knapsack* problem [82] to our problem. In absence of PUs, the problem is to simply say how many SU's requirement can be accommodated given the total available spectrum. Let's look at a very simplified version of the problem where all the subcarriers provide same data rate for all the SUs, i.e. in eq. 3.1, $R_{i,j} = r; \forall i, j$. Let's also assume that there is no cross channel interference i.e. $I_{i,j} = 0, \forall i \in \mathbb{C}, j \in \mathbb{C}; i \neq j$. All SUs have different data rate demand, $T_i^{req}, \forall i \in \mathbb{N}$. Each SU requires T_i^{req}/r subcarriers. Assume an SU can be allocated with spectrum if and only if its demand is met. Now, the goal is to maximize total throughput of the system, $\sum_{i\in\mathbb{N}} x_i T_i^{req}$ where $x_i \in \{0,1\}$ and $\sum_{i\in\mathbb{N}} T_i^{req}/r \leq C$. This problem is exactly same as the 0-1 knapsack problem. Since the simplified problem resemblance the 0-1 knapsack problem, we can say that the knapsack problem can be solved in polynomial

time if spectrum assignment problem can be solved in polynomial time. Since 0-1 knapsack problem is NP-hard and it can be mapped to our problem, we can conclude that the spectrum assignment problem is at least NP-hard. ■

3.1.3 Spectrum allocation methods

The proposed allocation methods are developed based on the optimization problem in eq. 3.6. In this research the spectrum allocation problem is addressed for three cases: 1) centralized spectrum allocation, 2) decentralized spectrum allocation where all SUs individually optimize spectrum utilization and 3) distributed SUs periodically perform coordinated spectrum defragmentation. Even though the current work assumes pilotless OFDM system, the same mechanisms can be applied to transceivers which do require pilot carriers by considering pilots' spectrum requirement along with the similar requirement of guard bands.

3.1.3.1 Centralized method

In this case one central controller or base station supervises the spectrum allocation. All SUs periodically sense the spectrum and send the spectrum usage map to the controller. Each SU also notifies the controller of changes in its throughput requirements (T_i^{req}). The network is assumed to use a dedicated out-of-band common control channel (CCC) [83] for transmitting control messages.

The controller has two states: *Steady* state and *Arrangement* state. It stays in the *Steady* state until a PU activity or change in an SU's requirement is detected, at which time it transits to the *Arrangement* state, where it invokes Algorithm 1 to calculate subcarrier allocation vectors: $\mathcal{A}, \mathcal{D}, \mathcal{G}$ for all SUs. The input to Algorithm 1 is the current PU usage vector and the throughput requirements of all SUs. The controller broadcasts the resulting $\mathcal{A}, \mathcal{D}, \mathcal{G}$ to all SUs and returns

to *Steady* state afterwards. The idea behind the Algorithm 1 is straightforward. Spectrum is allocated for the SUs with higher throughput demand with more priority, intending to reduce the number of fragments used by a single SU. An SU is allocated with spectrum if its requirement can be fully satisfied. First the SUs are sorted in the descending order of their throughput demand. Then for each SU, Algorithm 2 is used to allocate subcarriers. This is a recursive algorithm that explores all the possible combinations of allocating spectrum fragments to an SU and fix the combination that occupies minimum number of subcarriers. First it finds out the spectrum holes with the function *find holes*, which takes the channel usage vector and the subcarrier boundaries that an SU can use. This function returns list of usable holes considering guard bands, within the spectrum allocation boundaries of u_i. For each spectrum fragment, the required number of guard carriers is provided. If the demand is satisfied, it returns the number of allocated subcarrier considering guard bands, and also the number of fragments used. If the throughput demand is not satisfied, the spectrum usage boundary is updated and the function is recursively called. If for a combination one hole is left such that no other SU can use it due to guard bands, the remaining subcarriers are considered as wasted. This process is repeated considering all holes in descending order and then takes the combination that provides the lowest number of wasted subcarriers. If an SU cannot be allocated with spectrum, the function returns ∞. Algorithm 1 checks the returned value and if spectrum can be allocated for an SU, it updates $\mathcal{A}, \mathcal{D}, \mathcal{G}$ and moves on to allocate the next SU.

3.1.3.2 Distributed method

In this method the SUs are completely distributed. Hence central coordination is not possible. The receiver senses the entire spectrum in its aggregation range while receiving data. It then piggybacks the sensing information to the transmitter with acknowledgments (ACK) or other data packets. Figure 3.1 illustrates the

Algorithm 1: $Centralized\_allocation(\mathcal{A}_{PU}, \mathcal{T}^{req})$

input : A_{PU}, T^{req}
output: $\mathcal{A}, \mathcal{D}, \mathcal{G}$

$sorted\_list \leftarrow$ list of SUs in descending order of $\mathcal{T}^{req}$
for $i \in sorted\_list$ **do**
  $S, A, D, G \leftarrow channel\_assignment(\mathcal{A}_{PU}, low, high, T_i^{req}, i)$
  if $S \neq \infty$ **then** $A_i \leftarrow A$; $D_i \leftarrow D$; $G_i \leftarrow G$;
  else $A_i \leftarrow (0, ., 0)$; $D_i \leftarrow (0, ., 0)$; $G_i \leftarrow (0, ., 0)$;
  $A_{PU} \leftarrow A_{PU} + A_i$
return $\mathcal{A}, \mathcal{D}, \mathcal{G}$

Algorithm 2: channel_assignment(ch_usage, low, $high$, req, i)

input : Channel usage vector : ch_usage, Frequency lower limit : low, Frequency higher limit : $high$, Required throughput : req, SU: i
output: Number of fragments ∞ means unsuccessful allocation; channel allocation vectors: A, D, G

$hole \leftarrow find\_holes(ch\_usage, low, high, req, i)$
$H \leftarrow no\_of\_holes$
Sort $hole$ in descending order
$S \leftarrow (0, 0, \ldots, 0)$
Initialize A, D and G with $H \times C$ null matrices.
if *H=0* **then** Return $\infty, (0, .., 0), (0, .., 0), (0, .., 0)$;
if $req < 0$ **then** Return $0, (0, .., 0), (0, .., 0), (0, .., 0)$;
for *k :=1 to H* **do**
  for $c := hole_l^k$ *to* $hole_l^k + guard - 1$ **do**
    $A_{k,c} \leftarrow 1$; $G_{k,c} \leftarrow 1$
  for $c := hole_l^k + guard$ *to* $hole_u^k - guard$ **do**
    if $req > 0$ **then**
      $A_{k,c} \leftarrow 1$; $D_{k,c} \leftarrow 1$
      $S_k \leftarrow c$; $req \leftarrow req - R_{i,c}$
  for $c := S_k + 1$ *to* $S_k + guard$ **do**
    $A_{k,c} \leftarrow 1$; $G_{k,c} \leftarrow 1$
  $S_k \leftarrow S_k + guard$
  if $hole_h^k - S_k \leq 2 \times guard$ **then** $S_k \leftarrow hole_h^k$;
  if $req > 0$ **then**
    update low, $high$
    $S_c, A_c, D_c, G_c \leftarrow channel\_assignment(ch\_usage + A_k, low, high, req, i)$
    $S_k \leftarrow S_k + S_c$; $A_k \leftarrow A_k + A_c$
    $D_k \leftarrow D_k + D_c$; $G_k \leftarrow G_k + G_c$
    if $\neg constraints\_satisfied$ **then** $S_c \leftarrow \infty$;
$m = \operatorname{argmin} S$
return $S_m,\ A_m,\ D_m,\ G_m$

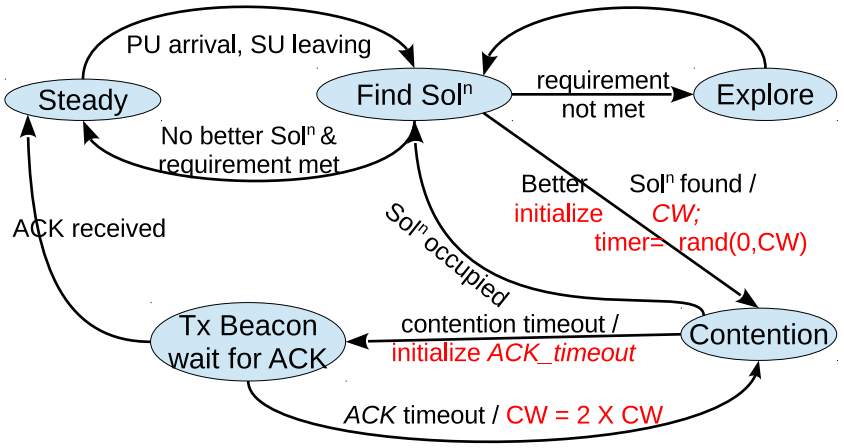


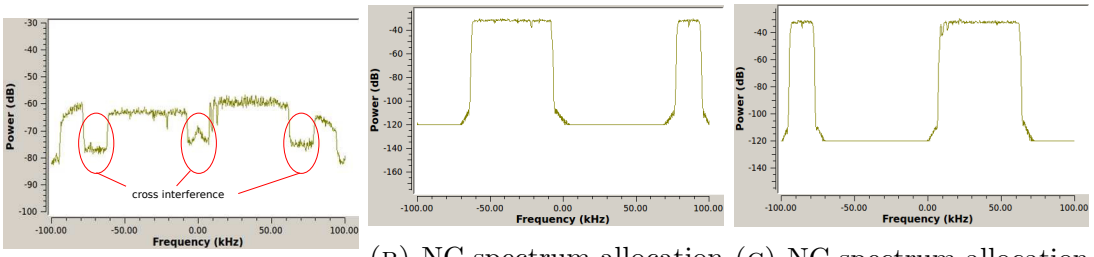
FIGURE 3.1: State diagram of an SU in complete distributed method

state diagram of an SU following this procedure. While in the *Steady* state, the SU transmits data over the subcarriers acquired earlier. If interrupted by a PU, it jumps to the *Find Sol*^{n} state to find another possible opportunity for its transmission. During the *Steady* state, if a new spectrum hole is observed, then it transits to *Find Sol*^{n} to trigger defragmentation. In *Find Sol*^{n} state, the SU calls Algorithm 2 to compute better subcarrier assignment opportunities. If the computed subcarrier assignment cannot facilitate the required throughput, the SU goes to the *Explore* state where it scans the spectrum outside of its current scanning range. If a better solution is found then the SU initializes the contention window (CW) and goes to the *Contention* state, where it begins counting down for contention timeout while monitoring the spectrum. If the SU senses another transmission on the targeted subcarrier before the timeout, it returns to *Find Sol*^{n}. If the desired subcarriers are idle throughout the contention period, the transmitter sends identifying beacons and waits for the ACK from the receiver. If the ACK is not received due to collision, the SU increases its contention window and goes to the *Contention* state. If the ACK is received successfully, it starts transmitting on the desired subcarriers and transits to the *Steady* state.

3.1.3.3 Semi-centralized method

The efficiency of the completely distributed method is not optimal since in such cases coordinated defragmentation or reassignment cannot be performed. On the other hand, the completely centralized method requires a dedicated controller. One idea is to combine both of these approaches to achieve a more efficient performance. But, in a distributed ad hoc network, committing individual resources to coordination is not desired by any user, since the central controller needs more computational power and spare time to coordinate the network. To overcome this issue, we propose a semi-centralized method. In this approach, all the SUs periodically defragment the spectrum through reassignment by a temporarily elected leader.

In this method, an SU follows the state transition of completely distributed method as discussed in Section 3.1.3.2 with an extra interruption. For periodic defragmentation, all SUs use a timer for the defragmentation cycle. The timer is initiated at the end of the defragment process. Upon timeout, an SU goes to *Leader election* where the network elects a leader in a cooperative manner. A leader can be elected with many different criteria such as the SU with the minimum amount of load, computational power, battery life, etc. or simply in round robin fashion. Leader election, integrity and the successful delivery of the control messages through CCC are well studied areas [84–86]. When an SU becomes the leader, it collects spectrum usage information as well as throughput requirements from all SUs. If the whole spectrum is not sensed, the leader initiates a cooperating sensing to scan the uncovered spectrum. The leader follows the centralized method described in section 3.1.3.1 to compute $\mathcal{A}, \mathcal{G}, \mathcal{P}$ and broadcasts them. After the broadcast the leader becomes a normal SU. If an SU is not acting as a leader it waits for beacons from the leader and sends its information to the leader. After receiving $\mathcal{A}, \mathcal{G}, \mathcal{P}$, if an SU finds that it has not been assigned spectrum, it goes to $Find\ Sol^n$ state



(A) Received signal (B) NC spectrum allocation for node A (C) NC spectrum allocation for node B

FIGURE 3.2: Power spectral density plots at receiver

otherwise it goes to *Steady* state and starts transmitting over the assigned sub-carriers.

3.2 System development

To investigate the feasibility of the proposed methods and relevant parameters of their practical implementation, a prototype based on a network of GNURadio [45] controlled Ettus-USRP B210 [46] Software Defined Radios has been developed. This network is formed of two pairs of transceivers as secondary users. Each node is an OFDM transceiver, capable of both contiguous and non-contiguous transmissions.

The radios were configured to operate over a 200 KHz band centered at 5.25GHz. OFDM transmitters and receivers divide this band to 256 subcarriers of 781.25 Hz each, which can be dynamically allocated to any user. For correct detection of OFDM preamble in each channel, a minimum of 28 subcarriers must be assigned to each transmission, which can be both contiguous and non-contiguous. Also, each radio has a spectrum sensing block, capable of sensing the entire aggregation range.

It was observed that, even after accounting for the frequency offset at the OFDM receiver, filtering alone is not sufficient for effective elimination of cross-channel interference, and some degree of frequency separation is required between adjacent

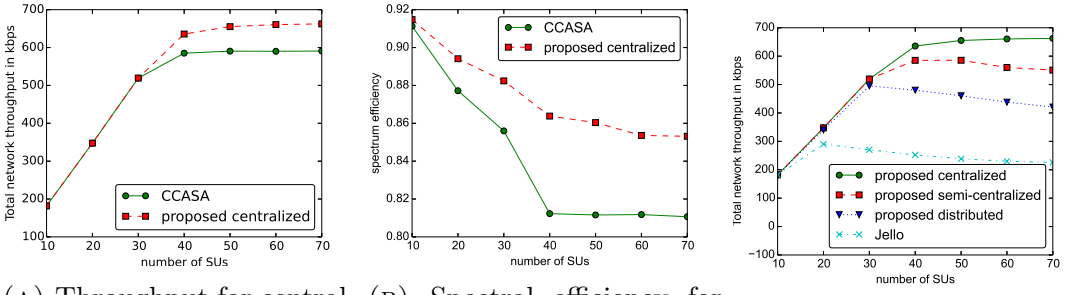
transmitters. Figure 3.2 depicts power spectral densities of two radios, and their concurrent Non-Contiguous OFDM transmission. From figure 3(a), it can be seen that when the two transmitters operate simultaneously, there is a considerable amount of interference in the frequency space between the two non-contiguous. Therefore, implementing guard bands between two adjacent transmitters is evidently necessary. To find the minimum number of required guard bands, the frequency separation between two transmitters using contiguous blocks was increased until the level of cross-channel interference fell below a threshold determined by the error rate in received data. Measurements indicate that the number of required guard bands varies with the level of received power. To achieve a practical number for necessary guard bands, multiple measurements with varying distances between antennas and minute changes in transmit power and receive chain gain were performed. It was then concluded that for the majority of the cases, given the fragile stability of USRP interfaces, a minimum of 2 guard bands is a practical choice, which is one of the considerations in simulations discussed in following sections.

3.3 Simulation results

The proof of concept testbed described in section 3.2 has limited capacity in terms of number of SUs, subcarrier aggregation range, etc. We developed a discrete event simulator in order to analyze the performance of the proposed method with broader spectrum range and more SUs. Necessary system parameters are obtained from the testbed in order to correctly evaluate the methods through the simulation. Table 3.1 provides the list of parameters used. Every simulation is run for *simulation time* and the simulator begins recording results after *Warm-up-time* to eradicate the fluctuations that occur in the early stage. The PU occupancy list and SU requirements list for the whole simulation time is generated at the beginning and identical copies are used for comparing multiple method. This whole

TABLE 3.1: Simulation parameters

Parameter	Symbol	Value
Simulation Time		$1,00,000$ *sec*
Warm-up-time		$10,000$ *sec*
replication		25
PU's width		$\sim U(10-20)$
PU active time		$\sim U(20,40)$ *sec*
PU sleep time		$\sim U(60,120)$ *sec*
Total subcarrier	$\mathbb{C}$	2048
Aggregation range	B_i	256
Data Rate per subcarrier	$R_{i,j}$	$\sim U(293,586)$ bps
Maximum fragmentation	F_i^{max}	10
no. of guard carrier		2
SU's requirement	T_i^{req}	$\sim U(10,30)$
SU's demand change interval		$\sim U(2,4)$ *sec*
Initial Contention Window	CW	8
ACK timeout		10 *msec*
central timeout		4 sec



(A) Throughput for centralized method (B) Spectral efficiency for centralized method (C) Throughput comparison

FIGURE 3.3: Simulation results

process is replicated several times and the values are averaged in order to obtain reliable results. In this simulation each PU has a different spectrum width and hence occupies a different number of subcarriers.

In the first stage, the performance of the centralized method is evaluated. We compare our centralized spectrum allocation method with CCASA [44]. While CCSA does not consider the waste of spectrum as a constraining parameter, our centralized algorithm takes the guard bands and pilot carriers into account in the calculation of the minimum spectrum requirement and allocation. Figure 3.3a plots the total throughput obtained for the network which is the sum of

throughput for all SUs in the network. It can be seen that, for both approaches, the average throughput increases linearly with increase in number of SUs present in the system. But it is also observed that in our centralized algorithm, the maximum capacity is significantly improved in comparison with CCASA. Figure 3.3b plots the spectral efficiency of the system. Spectral efficiency is defined as $\frac{\text{no. of allocated data carriers}}{\text{no. of allocated carriers}}$. The plot reveals that when there are more SUs contending in the system, the spectral efficiency is lower, which reveals the spectrum is heavily fragmented. It is also evident that, regardless of the number of SUs, the centralized algorithm performs better than CCASA in terms of spectral efficiency.

To assess the performance of our proposed methods, we compare them with the approach introduced in the Jello testbed [19], since it provides the most relevant results to ours, as it is developed based on similar fundamental systems assumptions, and also considers the problem of fragmentation. Figure 3.3c provides the total throughput achieved for the entire network. The results obtained from all of our methods indicate a significantly better performance in comparison with Jello. As the number of contending SUs increase, the probability of collision increases during the contention period, which results in the fall of the average throughput of the network. When the spectrum aggregation range is much smaller compared to the total spectrum width, the proposed semi-centralized method performs significantly better than Jello. One reason for this enhancement is that Jello looks for spectrum in a fixed frequency range, while our methods are capable of looking for holes anywhere in the frequency range supported by the transceivers using a sliding frequency window approach. The proposed distributed method obtains 42.43% higher throughput compared to Jello. It is also noteworthy that the semi-centralized approach does not perform as well as the completely centralized algorithm, but the semi-centralized method achieves a 27.14% improvements over the proposed distributed method.

3.4 Summary

In this chapter we briefly described the challanges in dynamic spectrum acquirement and how we can optimize the spectrum acquirement. Online Defragmentation was proposed as a method of increasing spectrum utilization in channel-aggregating DSA radio networks. Efficiency of this method was investigated in three different network scenarios:Infrastructure, distributed and semi-centralized. By including parameters retrieved from a proof-of-concept prototype into simulations, realistic comparisons of the three scenarios with regards to effectiveness of the presented algorithm have been presented. It was concluded that regardless of scenario, defragmentation provides better performance in terms of spectral efficiency and throughput.

Chapter 4

CR-Honeynet

In the last chapter, we discussed how a group of SUs can acquire spectrum optimally. Here each SU is assigned with a spectrum which it can use for transmitting data. Now we focus on the challenge of survivability under adversarial conditions. With the advent of "smartness" and "learning" in the cognitive radios, the challenge of survivability will not be trivial. An adversary can disrupt the communication of a legitimate SU by transmitting on the same spectrum. However, from an intelligent and rational attacker's perspective, jamming a communication randomly will not yield optimal results. Rather, an attacker can be most disruptive if it targets the communication that impacts the CRN most severely upon interruption [15, 20–22].

In this chapter we present *CR-Honeynet*, a honeynet based defense mechanism where the CRN passively learns the strategy of the attacker using stochastic learning, and then places an active decoy, namely *honeynode*, to entice the attacker into jamming the honeynode transmission. Thus, the attacker gets a false impression of attacking the highest impacting communication whereas legitimate SU communications avoid attacks and reducing attack impact on the CRN. One or multiple SUs act as honeynode in each transmission period. The SU acting as honeynode

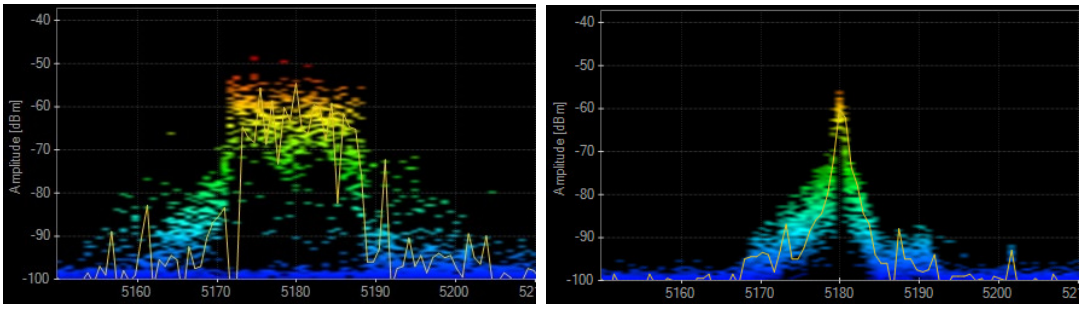
refrains from transmitting its own data packets and transmits garbage data with specific transmission characteristics. Such transmission characteristics lure the attacker to jam honeynode's transmission. The transmission characteristics that the attacker employs is learned from the history of attacks. As an example, if an attacker targets highest transmission power then the honeynode transmits with highest possible power while all other SUs keep their transmission power lower than honeynode's power.

The evolving nature of the attacker, as well as dynamic and stochastic nature of the wireless medium pose several challenges to the learning mechanism. Suspicious of being trapped, an attacker may intentionally change its strategy of finding highest impacting communication. Also, due to erroneous and stochastic nature of wireless medium, an attacker may err in sensing CRN's highest impacting communication. Such error may results in attacks on different SU communications instead of the communication with desired/targeted characteristics. Such circumstances must be taken into account for effective luring. In this research, we use statistical monitoring threshold to decide whether the change in recent attack patterns is due to an error in the attacker's sensing, or whether the attacker has changed its attacking strategy. Our proposed stochastic learning mechanism correctly detects an attacker's strategy with a probability of 0.958 within 15 iterations and identifies change in attacker's strategy dynamically with 95% confidence interval within 5 iterations. The simulation results show that CR-Honeynet learns an attacker's strategy correctly and adapts as an attacker's strategy changes dynamically, which in effect enhances CRN's performance in terms of packet delivery ratio. We further develop a state-of-the-art testbed using off-the-shelf software defined radios which support the effectiveness of the defense mechanism proposed.

To protect PU incumbent services, DSA strictly enforces SUs to periodically pause its transmission and sense for PU activity [87]. SUs scan the wireless environment for free channels in the *sensing period* and transmit packets during *transmission*

period. The centralized controller allocates different channels to each SU. Several practical challenges need to be co-opted and addressed before allocating resource for honeynet in CRNs. Although dedicating an SU as honeynode potentially makes the CRN robust, it is not a "free ride" as it degrades the effective system throughput. A critical question is how would the honeynode be chosen then? Who will be responsible ("honeynode" selection) for auxiliary communications and monitoring in honeynode? To answer the above questions, we must first understand the complexity of the CRN's traffic behavior under DSA scenario. Consider a scenario wherein a user is conducting a number of simultaneous transmissions for example, videoconferencing. All theseThese types of applications generate packets randomly and independent of other applications. The complex nature of data traffic makes it difficult to analyze the Quality of Service (QoS). CRNs, meanwhile, exhibit a unique behavior pattern that remains yet to be investigated by any mathematical model. For example, the periodic sensing by SUs forces interruption on transmission, affecting end-to-end QoS by imposing delay and jitter on packet transmission. Thus, a major goal of this work is to model a CRN's service using stochastic analysis and use our model to estimate baseline performance indicators. Then we propose state dependent honeynode selection policies for different traffic models to enhance the CRN's performance.

The rest of the chapter proceeds as follows: in Section 4.1, we discuss the motivation for our work by demonstrating the severity of jamming with protype system and how the concept of honynet can be used. Section 4.2 presents our proposed CR-Honeynet model. In section 4.3 we describe initial simulation how CR-Honeynet learning phase works and its performance. The prototype system and the experiments are described in Section 4.4. Section 4.5 presents a mathematical model to estimate CRN performance using a queue with fixed periodic server vacation. Section 4.6 presents several honeynode selection policies. In Section 4.7, we build a comprehensive simulator to study the performance of the proposed



(A) Normal communication (B) Jamming Signal

FIGURE 4.1: PSD for data communication and jamming signal

model, describe a utility model to determine when a honeynet can be used and when not, measure the fairness of all honeynode selection strategies and finally present the benefits of an optimal honeynode selection strategy that provides the best performance with fairness. Finally, Section 4.8 summarizes the chapter.

4.1 Motivation

4.1.1 Jamming attack

The threat of penalty can discourage a potential assailant from attacking a PU; however, when an SU accesses a channel, it borrows the channel, and it does not have any ground from which it can fend off attackers. While PUs are able to discourage attackers, SUs are left vulnerable to malicious jamming / disruptive attacks [88]. Jamming can be broadly categorized into two types [76, 89]. The first type being *physical layer jamming* where the attacker jams the channel of communication by sending strong noise or a jamming signal and the second type is *datalink / MAC layer jamming*, which targets several vulnerabilities present in the MAC layer protocol.

We run an experiment in our lab, to study the feasibility of jamming attack. Two computers are configured to communicate over a WLAN (IEEE 802.11-a,

channel 36, central frequency: 5.18 GHz). The Power spectrul Density (PSD) for normal communication (without jamming), which can be seen in Figure 4.1a is observed through Wi-spy spectrum analyzer [90]. Then we begin transmitting a narrow-band jamming signal of 2MHz from a GNU Radio [45] enabled USRP radio [46] on the same channel. In the presence of the jamming signal, the genuine transmission of the WLAN was stopped completely which can be observed in Figure 4.1b. Here, the attacker is exploiting the vulnerability present in IEEE 802.11 MAC that enforces a node to sense the channel before transmission. When the legitimate transmitter senses that there is some energy on the channel, it refrains from transmission. Irrespective of the jamming technique, a target node suffers significant amount of data or packet loss and sometimes completely loses the channel. CRN, being a next generation intelligent network, should incorporate a mechanism to mitigate, avoid, and prevent such attacks.

4.1.2 Use of Honeynet in avoiding attacks

"Honeypot," in cybercrime, is defined as "a security resource who's value lies in being probed, attacked or compromised". In cybercrime defense, honeypots are being used as a camouflaging security tool with little or no actual production value to lure the attacker into giving them a false sense of satisfaction, thus bypassing (reducing) the attack impact and giving the defender a chance to retrieve valuable information about the attacker and their activities. This node is called honeynode. A single channel honeypot-based channel surfing, to mitigate jamming-based DoS attacks, has been proposed in [89]. The network dedicates a node, as honeypot, to monitor attacks. Upon detection of attack, the network switches its channel of operation, which results in long-time communication disruption. Majority of the previous works have assumed that the attacker is naive and does not evolve. Thus, none of these works have focused on learning the strategy of attacker where

the attacker is also dynamic and changes its strategy of choosing the target communication characteristics.

From an intelligent and rational attacker's perspective, jamming a communication randomly will not yield optimal results; rather, an attacker can be most disruptive if it targets the communication that impacts the CRN most severely upon interruption [15, 20–22]. The attacker succeeds in determining highest impacting communication by observing certain transmission characteristics, for example, highest transmission power, highest data rate, modulation scheme, packet inter arrival time, quality of route with end-to-end acknowledgments, etc. [21]. To proactively defend against such intelligent attackers, a CRN must learn about the strategy through evolution [91, 92] that the attacker uses, to figure out the highest impacting communication. The attacker's strategy of finding the highest impacting communication can be used as a trap by the defending CRN to detract the attacker from striking legitimate communications.

We propose *CR-honeynet*, a honeynet-based defense mechanism where the CRN passively learns the *strategy of the attacker* and then places an active decoy, namely *honeynode* to entice the attacker for jamming the honeynode transmission. Thus, the attacker gets a false impression of attacking the highest impacting communication, whereas legitimate SU communications avoid attacks and reduce attack impact on the CRN. The SU, acting as honeynode, refrains from transmitting its own data packets and instead transmits garbage data with specific transmission characteristics. Such transmission characteristics lure the attacker to jam the honeynode's transmission. For example, if an attacker targets the highest transmission power, then the honeynode transmits with highest possible power, while all other SUs keep their transmission power lower than the honeynode's power. The description of the learning mechanism of honeynet is provided in Section 4.2.

4.1.3 Special queue model

When CR-honeynet is deployed, and the attacker is evolving, the attacks will sometimes be trapped by honeynode, and sometimes can strike legitimate SUs. We define one parameter, *attractiveness of honeynet* (ξ) as the probability that the honeynode is the one to be attacked, conditional on observing a jamming attack. Note that ξ depends on how well the CR-honeynet learning mechanism works. Our next goal is to investigate the effectiveness of CR-honeynet with different values of ξ and determine when it is/not beneficial to deploy CR-honeynet.

Honeynode ensures less data loss at the cost of end-to-end delay. Some application can tolerate data loss but not delay and others the opposite. The goal is to build a mathematical model that can estimate system performance before we actually deploy CR-honeynet. If honeynode assignment results in degradation of overall system performance then we can opt for not assigning honeynet. The end-to-end delay in CR is mainly affected by queuing delay as processing, transmission and propagation delays are negligible compared to queuing delay. Our theoretical model focus on determining queuing delay. Then we concentrate on honeynode selection strategies to achieve better over-all system performance.

We can model an SU as a server with vacation where vacation is special service with higher priority. There are many mathematical models that deals with servers with vacations [93–96] where the server has the option to take vacations only at the end of its current service. Because the sensing period has deterministic length and intervals, the server model does not conform to the usual server with vacations. Instead, the sensing period acts as a "priority" customer whose inter-arrival rates and service times are deterministic. In this case a packet transmission has to be delayed if this packet can not be transmitted within the transmission period. In Section 4.5 we derive a mathematical model for server with deterministic vacations that portray the behavior of SUs in CR-Honeynet.

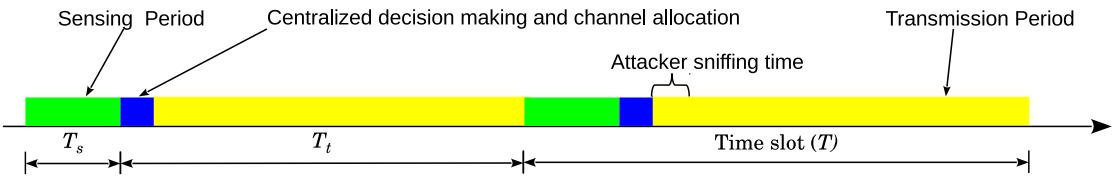


FIGURE 4.2: Time slots in cognitive cycle

4.2 CR-Honeynet

4.2.1 Assumptions

i) **Network:** The network consists of one central controller and several SUs. The central controller oversight the channel used by SUs.

ii) **Time Slots:** To protect PU transmissions, SUs are required to perform periodic spectrum sensing and evacuate promptly upon the return of the PU. A time slot consists of a sensing period followed be a transmission period as can be seen in Figure 4.2.

iii) **Sensing Period:** SUs scan the wireless environment for free channels in the *sensing period* (T_s). SUs send the spectrum usage report to the central controller for decision.

iv) **Transmission period:** During transmission period (T_t), a SU transmits packets through its own channel dedicated by a centralized controller.

v) **Jammer:** Jammer is itself an SU with the same power of a normal SU.

vi) **Attack:** We assume that a SU cannot switch its channel during the transmission period as it is unaware of the condition of the other channels and can switch only on the next transmission period. Upon being attacked, all data packets transmitted by the SU are lost.

vii) **Common control channel:** The network uses a out of bound common control channel for control message communication between the central controller and the SUs. This channel is proprietary and the attacker can not hear decrypt these messages.

4.2.2 Model for attackers

In this research we consider three types of attacking strategies, as follows:

I: Attacker targeting a particular channel.

II: Targets specific SU transmission characteristic(s).

III: Randomly targeting channel with active SU transmission.

Attacking strategy of type I and III causes less harm on a CRN as it does not search for the best communication that causes the highest impact in CRN. However, an intelligent and rational attacker of type II can choose any transmission characteristics to determine the best communication for attack that causes highest impact on the CRN. From the CRN's point of view, it is difficult to generate such characteristic space. Such targeted characteristic space of an attacker can be learned by two methods: *manually by domain experts* or through *automatic learning* from data obtained for long time. For the first step, we are dealing with the first method and wish to extend our model to perform the second option and learn an attacker's possible strategical view points by automatic learning. We present a generalized model considering the d possible transmission characteristics or a combination of transmission characteristics.

4.2.3 CR-Honeynet defense mechanism

In CR-Honeynet, the central controller assigns the role of honeynode to a SU at the beginning of each transmission period. Figure 4.3 illustrates this channel allocation based on time domain (Sensing period not shown). Due to the error of the attacker or strategy change of the attacker, some attacks are trapped by honeynode transmission, and others disrupt legitimate SU communications. We define a parameter, *attractiveness of honeynode* (ξ) as the probability that the honeynode transmission is attacked, conditional on observing a jamming attack.

When acting as a honeynode, a SU doesn't transmit its packets, instead, it queues all its incoming packets and transmits garbage data packets. Honeynode allocation results in more delay as well as packet drop due to finite buffer sizes for the chosen SU, both of which are undesirable. If the attractiveness of the honeynet (ξ) is low, then the CRN will suffer the delay caused by honeynode allocation as well as the packet drop with a probability of $(1 - \xi)$ due to the attacks on legitimate SUs other than the one chosen as honeynode. The threshold, lowest attractiveness of the honeynet (ξ^*) is the value where the net gain is zero; below ξ^* the CRN is better off facing the loss from the attacks than dedicating one SU intending to lure the attacker.

In accordance to an attacker's strategies, we define the following honeynode strategies:

i: If the attack strategy is believed to be of type-I then the vulnerable channel will be assigned to the honeynode.

ii: If the attack strategy is believed to be of type-II then the actual target property should be learned and used as a lure for the honeynode.

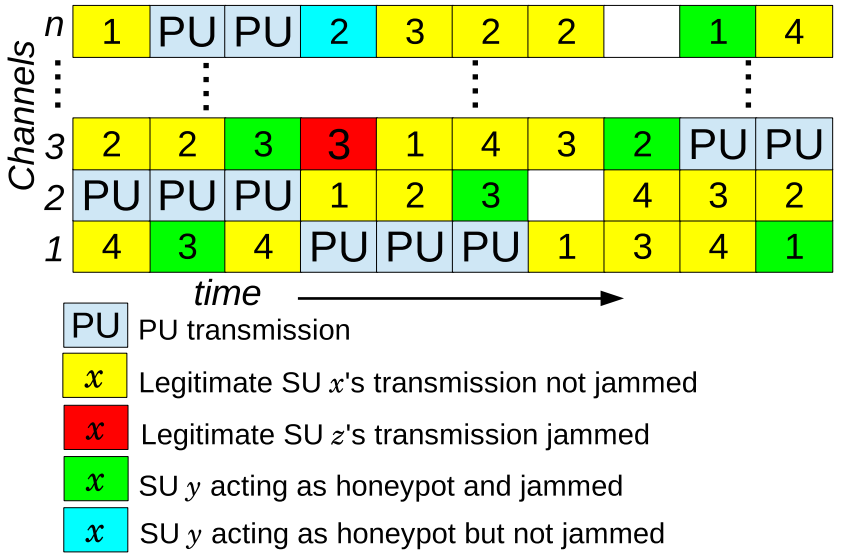


FIGURE 4.3: A snapshot of CR-Honeynet channel allocation

iii: If the attack strategy is believed to be of type-III then we use a *special honeynode strategy* that delays all but the honeynode's transmissions in order to reduce the number of vulnerable channels to 1.

If there are $\mathcal{C}$ available channels, then an attacker must sense for activity on each of them. Let's assume switching a channel incurs a delay of κ ($\kappa = 7.6ms$ has been measured for Atheros WiFi [75]). The attacker needs at most $\mathcal{C}\kappa$ time units to scan all available channels for activity. Under the special strategy we must therefore delay all SUs at least $\mathcal{C}\kappa$ units of time beyond the sensing period, during which only the honeynode will transmit. When using the special strategy there is an added loss or cost of luring as all the other SUs are delayed in their transmissions, albeit much less than the delay caused to the chosen honeynode. So this strategy should be avoided by CR-Honeynet if possible. In contrast, type-III is the only strategy that increases attractiveness of honeynet (ξ) to 1.

4.2.4 Stochastic model

We assume that at time slot n the attacker's strategy S_n follows a random switching process, with consecutive switching times $T_k \in \mathbb{N}$. The model need not be a

Hidden Markov Model, but we assume that the holding times $h_k = T_{k+1} - T_k$ are long enough for learning. We will specify the exact assumption later on.

The *base model* for an attacker with a type-II strategy is stated now. Because of measurement errors, the attacker may not always be successful in identifying the correct communication to attack. Let p_1 denote the probability of attacking the communication with the target characteristics. We will assume that the number of available channels is larger than d, and use d of the SUs as learning probes, each with a different target property. Counting only the time slots when one of the probes is attacked, the total number of attacks to each of the probes within n such time slots is modeled as a multinomial random variable with probabilities:

$$p_1(\theta) = \frac{\theta}{\theta + d - 1}; \quad p_i(\theta) = \frac{1}{\theta + d - 1}, \text{ for } i \in \{2, 3, \ldots, d\}, \tag{4.1}$$

where $\theta > 1$.

The above model corresponds to the situation where probe $k = 1$ is targeted and hit with probability $p_1 < 1$. Under error measurement, any other probe will be attacked with equal probability $p_i, i \neq 1$. The number $\theta = p_1/p_i$ provides the ratio between p_1 and the rest. For the base model, using the fact that all other probabilities are equal, $\theta = (d-1)p_1/(1-p_1)$.

Define the function:

$$\phi(\theta, n) = \sum_{y \in \mathcal{P}(n)} \frac{n!}{y_1!\, y_2!\, , \ldots y_d!} p_1(\theta)^{y_1} \left(\frac{p_1(\theta)}{\theta}\right)^{\sum_{i=2}^{d} y_i} \tag{4.2}$$

where the summation is over the set of all possible observations of a sample of size n of the multinomial with parameters eq. 4.1 where the first component dominates the others, that is:

$$\mathcal{P}(n) = \left\{ y \in \mathbb{N}^d : \ \sum_{k=1}^{d} y_i = n, \text{ and } y_1 \geq y_i; i \geq 2 \right\}.$$

It is straightforward to show that this is the exact probability of correct selection in a sample of size n from the base model when the maximum likelihood estimator is used. Specifically, let $Y_i(n)$ count the number of attacks to probe i under the base model, so that: $(Y_1(n), Y_2(n), \ldots Y_3(n)) \sim M(p(\theta), n)$, then the MLE for the parameter p_i is simply $\hat{p}_i(n) = Y_i(n)/n$ and $\phi(\theta, n) = \mathbb{P}(Y_1(n) = \max(Y_1(n), \ldots, Y_d(n))$.

Let $\alpha \in (0, 1)$ be a confidence level for statistical significance. Then under the base model we can calculate the sample size required to ensure a probability of correct selection of at least $1 - \alpha$:

$$N^*(\theta, d) = \min\left(n : \ \phi(\theta, n) \geq 1 - \alpha\right). \tag{4.3}$$

Bechhofer *et al.*[97] have tabulated the function $\phi(\theta, n)$ for $d = 2, 3, 4$ using various values of θ and n. For example, if $d = 4$, then a sample size of $n = 25$ ensures a correct selection with level $\alpha = 0.200579$ when $\theta = 2$, and with level $\alpha = 0.038559$ when $\theta = 3$.

Suppose that honeynet correctly identifies a lure, but $p_1 < \xi^*$. Clearly, the best it can do here is to use its (correct) guess for the honeynode, but this will provide at most a probability p_1 that the honeynode will be attacked. Because p_1 is below the threshold, it will not be worth using honeynode in this case and we use special honeynet strategy similar to type-III. Thus, such values of ξ^* provide a threshold value $\theta^* = \frac{(d-1)\,\xi^*}{(1-\xi^*)}$ below which it is not worth using honeynode.

Definition: We call a *naive* attacker one of type II where the probability of error in measurement is lower than $1 - \xi^*$, and we assume that $\mathbb{P}(h_k < N^*(\theta^*, d)) = 0$.

The above definition says that this type of attack is fairly accurate (usually $p_1 \gtrapprox .85$) and also that the strategy is kept long enough to learn the target probe.

Specifically, because $\theta \geq \theta^*$ for a naive attacker, then $N^*(\theta, d) \leq N^*(\theta^*, d)$ if we use the MLE to identify the target with $\arg\max(Y_i(n))$ for $n \approx N^*(\theta, d)$.

4.2.5 Learning attacker's strategy

When the learning mechanism starts, given a confidence level α, the number $n = N^*(\theta^*, d)$ is calculated as a first estimate for an adequate sample size to detect type II attackers. When $d < \mathcal{C}$ it is possible that error in measurements results in false attacks to communications that have not been allocated any lures. Thus, we will focus only on time slots when attacks happen to lures. According to our model, this "sampled" process corresponds to the base model for attackers of type-II. Given n, define $\tau(n)$ as the total number of time slots required to see n attacks to the lures.

During the learning phase, the d different lures for type-II attacks are assigned to d different communications among the available ones with uniform probability and no honeynode is yet allocated. Let $Y_i(0) = 0; i = 1, \ldots, d$ and define for each $i = 1, \ldots, d$ and the counting processes:

$$Y_i(k) = Y_i(k-1) + \mathbf{1}_{\{i\text{-th lure is attacked at time } k\}}$$

for $k = 1, 2 \ldots$, where the notation $\mathbf{1}_{\{A\}}$ stands for the indicator function of event A (or Dirac delta). In parallel, define $C(1) = c$, if c is the first channel to suffer an attack, and let

$$C(k+1) = C(k)\mathbf{1}_{\{\text{channel } c \text{ is attacked at time } k+1\}}.$$

Because we have allocated the lures randomly among SU communications, it follows that

$$\mathbb{P}(C(k) = 1 \,|\, \text{type II or III}) \leq \max\left\{\left(\frac{1}{d}\right)^k, \left(\frac{1}{\mathcal{C}}\right)^k\right\}$$

Define n_0 as the smallest power that makes this probability smaller than our given confidence level α, that is, when $d < \mathcal{C}$

$$n_0 = \lceil \log(1/\alpha) - \log(d) \rceil.$$

The number of tests to check for type I is thus typically very small. For example, if $\alpha = 0.001, d = 2$ then $n_0 = 7$, for $\alpha = 0.005$ and $d = 6$, $n_0 = 4$.

If $C(n_0) = c$, we declare having learned that the attack is of type I and we identify c as the target channel. From this point onwards, we place the honeynet in this channel and keep monitoring. Because attacks of type I are not subject to error in identifying the channel, as soon as $C(k) = 0$ we declare a regime change and re-set the learning phase.

Otherwise, if $C(n_0) = 0$ then we keep assigning lures to channels for as many time slots are required to observe $n = N^*(\theta^*, d)$ attacks to lures. Gelfand *et al.*[98] provides a comparison between various estimators and confidence intervals for $\hat{p}_1$. In particular, his findings support the fact that under attacks of type II the approximate confidence interval based on the CLT is adequate, even for small to moderate sample sizes. Following this approximation, if

$$\hat{p}_1 - 1.96\sqrt{\frac{\hat{p}_1(1-\hat{p}_1)}{n}} \geq \xi^* \tag{4.4}$$

then we declare having learned that the strategy is of type II and we identify the lure. From this time onwards, we use the honeynode with that lure and start the monitoring phase. Notice that by construction, naive attacks are ensured to be correctly identified with probability at least $1 - \alpha$.

If eq. 4.4 does not hold, then we do not have significant evidence that our candidate lure will be sufficiently effective. From this point onwards (whether the attacker is of type II but with large measurement errors, or of type III) we use the special honeynode allocation by delaying all other SUs. It is important to note that while the regular honeynode entails a delay for the chosen SU, the special strategy delays all of the rest of the SUs, albeit by a much smaller amount of time.

4.2.6 Regime change detection: monitoring phase

Once the learning period is over, the corresponding honeynet strategy is used and honeynet keeps monitoring the attack counts, keeping track of running window averages. This is the monitoring phase where the honeynet is sensing for a possible change in attacker's strategy, as follows.

If the honeynet is under the assumption of a type I attack, then it keeps track of $C(k), k \geq n_0$ until the first time slot where $C(k) = 0$. Then it restarts the learning phase.

If the honeynet is under the assumption of a type II naive attacker then it uses sliding window averages to test for regime changes. During the monitoring phase the honeynet uses the detected lure for the honeynode allocation. Honeynet's first monitoring test uses a standard control chart for frequencies, and the second proposed method uses a regression for the slope of the frequency of attacks. Let $\hat{p}_1(k); k \geq n$ be as in eq. 4.5 re-calculated with increasing observations beyond the initial horizon n and call

$$L(k) = \hat{p}_1(k) - 3\sqrt{\frac{\hat{p}_1(k)(1-\hat{p}_1(k))}{w}}.$$

Given a window of size w time slots, let $\tilde{\xi}_w(k)$ be the estimate of p_1 (and also of ξ) for time slot $k > n$ using the observations $(Y_{(d)}(k-w), \ldots, Y_{(d)}(k))$. As soon as

$\tilde{\xi}_w(k) < L(k)$, the honeynet declares a change of regime and restarts the learning phase (resetting all counters).

The regression test works very similarly. (To complete, regression with the window and test for $H_0 : \beta < 0$, where β is the slope or method of residuals).

Finally, if the honeynet is operating under the assumption of a type III attack or a type II attack for small ξ, then honeynode's current strategy is the special honeynet strategy that delays all but the honeynode. The honeynet keeps a new counter $H(k) = H(k-1) \times \mathbf{1}_{\{\text{honeynode is attacked}\}}$, initialized at the value 1. As soon as the attack goes to another channel ($H(k) = 0$), the honeynet declares that the attacker is not aiming at random, but it must be targeting now either a specific channel or a specific property of the transmissions. Then the honeynet restarts the learning phase.

4.3 CR-Honeynet learning phase

4.3.1 Simulator

We coded a *tick based simulator* [99] using *Python* for simulating the CR-Honeynet. In the simulation we have considered 20 SUs and 1 attacker which can effectively attack one SU communication. The CR-Honeynet dedicates 1 SU as honeynode in each slot. The attacker follows Algorithm 3 and the honeynet follows Algorithm 4. All SUs generate packets in accordance with *Poisson* process and queue them while in sensing period or when that SU is acting as a honeynode. During transmission period, SUs that are not acting as honeynode transmit packets from the queue. Packet transmission time (S_n) follows uniform distribution of 0.1 - 1.7 ms. A sensing Period (T_s) of 50 ms and a transmission Period (T_t) of 950 ms has been

considered for cognitive cycle. We consider attacker has target transmission characteristics (d) space as 4. From the CRN's point of view, attractiveness threshold (ξ^*) is considered as 0.6. Type-I learning horizon (n_0) and Type II learning horizon ($N^*(\theta^*, d)$) are calculated as 5 and 15 respectively. We have run the simulation for 5,000,000 ms *simulation time* with 100,000 ms as *warm-up time* [1].

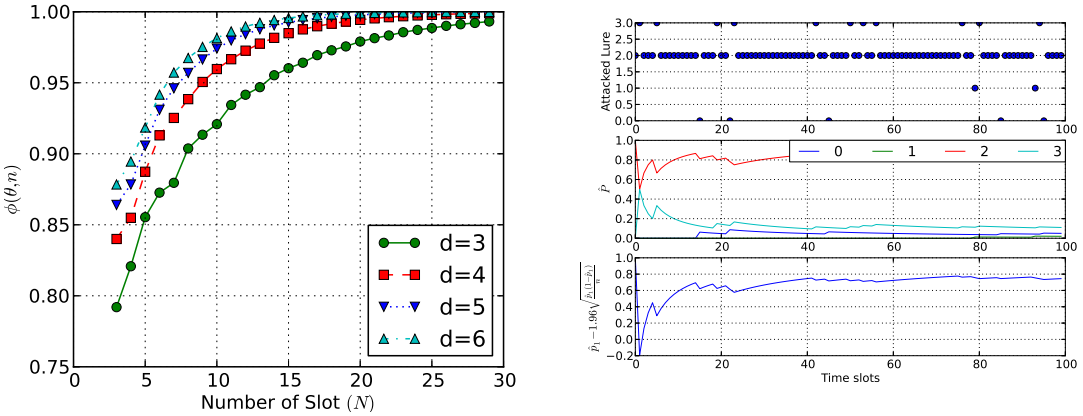
Algorithm 3: Algorithm for attacker

if *strategy = attack particular channel* **then**
  scan channel $c \in \mathcal{C}$ in the initial stage of T_t
  if *SU is active on c* **then**
    attack on channel c
else if *strategy = attack transmission characteristics x* **then**
  Scan all $c_i \in \mathcal{C}$ at initial stage of T_t
  attack the channel which have highest x
else if *strategy = attack randomly* **then**
  Scan all $c_i \in \mathcal{C}$ at initial stage of T_t
  attack randomly a channel c where SU is active

Algorithm 4: Algorithm for Honeynet

Calculate n_0, N^* based upon d and ξ^*
Reset all counters such as y, n etc.
Run initial learning phase for n_0 slots
if *all attack happens on channel* $c \in \mathcal{C}$ **then**
  Put honeynet on c in every slot until attack observed on other channel.
  Go to step 2
else
  Continue counting for $n = N^*(\theta^*, d)$ slots
  if $\hat{p}_1 - 1.96\sqrt{\hat{p}_1(1-\hat{p}_1)/n} \geq \xi^*$ **then**
    $lure = \text{argmax}(y)$
    put honeynode with $lure$ on every slot until $\xi_w(k) < L(k)$
    Go to step 2
  else
    Use special honeynet strategy until honeynode is not attacked. Go to step 2.

[1] To obtain reliable steady state results for system staring with empty queue, a simulator run for *warm-up period* [99] without recording data. Once the warm period is over simulator starts gathering data



(A) Confidence level of Learning

(B) Depiction of Learning while $d = 4$

FIGURE 4.4: Pilot simulations

4.3.2 Learning attacker's strategy

We plot $\phi(\theta^*, n)$ (eq. 4.2) with respect to learning period (N^*) in Figure 4.4a. We can clearly see that with an increase in N, confidence level also increases. We have defined earlier, confidence level for statistical significance ($\alpha = 1 - \phi$). From eq. 4.3 we can get the optimal N^*. For our simulation we have considered $\xi^* = 0.6$. We see that $N^* = 15$ ensures correct learning with level $\alpha = 0.015$ for $d = 4$. From the figure we can conclude that for a certain desired confidence of learning ($\phi(\theta^*, n)$), an increase in the number of lured characteristics (d), results in a decrease in required slots for learning (N^*). In another way, the more transmission characteristics or combination of transmission characteristics an attacker can target, CR-honeynet takes lesser time to learn with the same confidence.

Figure 4.4b provides an illustration of a learning phase. In this scenario, the attacker with type II strategy is aiming for lure 2 (the lure is characteristics of transmission) to attack. Lure 2 is actually attacked with a probability 0.8. The actual attacks on the various lures are shown on the upper subplot. The middle subplot depicts $\hat{P}$ for the different lures calculated using eq. 4.5. The third subplot provides $\hat{p}_1 - 1.96\sqrt{\frac{\hat{p}_1(1-\hat{p}_1)}{n}}$ which can be taken as a measure of learning. With

$d = 4$ and $\xi^* = 0.6$, the probability of correct selection after $N^* = 15$ samples is 0.985.

4.3.3 Dynamic evolution with change in attacker's strategy

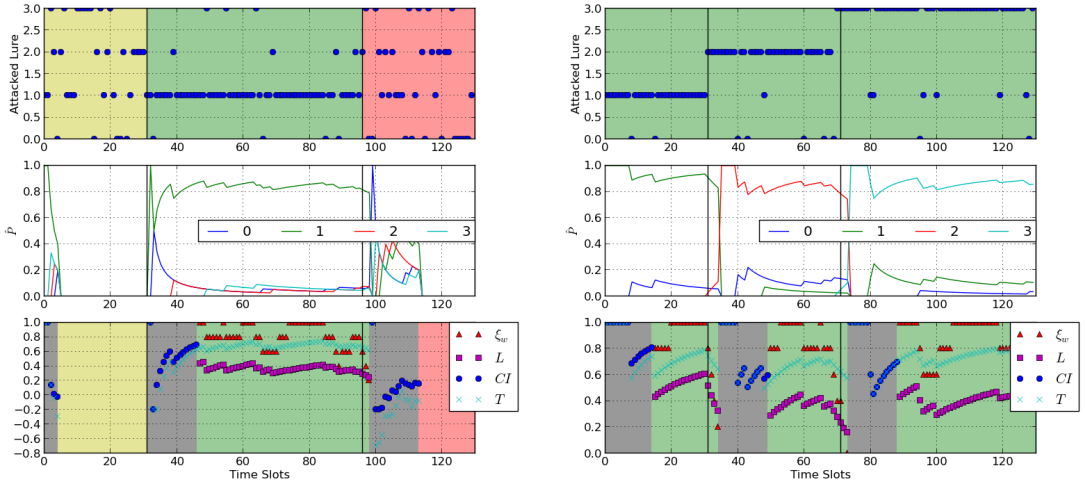
Figures 4.5a and 4.5b provide the results for different experiments, each of which corresponds to a different sequence of attack processes $\{S_n; n = 1, 2, \ldots\}$. The upper subplots provide the attacker's strategy. *Yellow, green* and *red* colors indicate type-I, type-II and type-III attacking strategies respectively. Blue dots indicates the attacker's aimed transmission characteristics to find highest impacting communication. Here we have used 4 types of lure, i.e. transmission characteristics $(d = 4)$.

Middle subplots give CR-Honeynet's observation of $\hat{p}$ eq. 4.5 for different lures. It uses the MLE estimators for the two highest probabilities:

$$\hat{p}_1 = \frac{Y_{(d)}(\tau(n))}{n}; \quad \hat{p}_2 = \frac{Y_{(d-1)}(\tau(n))}{n}, \tag{4.5}$$

where the notation $(x_{(1)}, \ldots, x_{(d)})$ is the usual notation for the ordered statistics.

In lower subplots of these 3 figures, we present phases of Honeynet. Background colors *Grey, yellow, green* and *red* indicate the learning phase, type-I, type-II and type-III defense strategies respectively. Then we plot the estimation of $CI = \hat{p}_1 - 1.96\sqrt{\frac{\hat{p}_1(1-\hat{p}_1)}{n}}$ in the learning phase. We can see that with increase in slots, CI is increasing. When the learning phase is over and honeynet decides which strategy to take, it change its phase. When it detects a regime change, it enters to the learning phase again. When it is in type-II honeynet strategy, the honeynet monitors $L(k)$ and $\tilde{\xi}_w$. Honeynet enters learning phase when $\tilde{\xi}_w \leq L(k)$. An approximate test of level 0.05 which decides whether the attack is of type II or III



(A) Honeynet Learning phase corresponding to attacker's strategy change from I to II and then to III

(B) Honeynet Learning phase corresponding to attacker of type II and attacker is changing its target lure

FIGURE 4.5: Simulation results

is to test if $T > 0$, for the statistics:

$$T = (\hat{p}_1 - \hat{p}_2) - 1.96\sqrt{\frac{\hat{p}_1(1-\hat{p}_1) + \hat{p}_2(1-\hat{p}_2) - \hat{p}_1\,\hat{p}_2}{n}}.$$

If $T \leq 0$ then we infer, the attacks are "sufficiently random" between at least two main contenders.

Figure 4.5a shows how the honeynet learns the change of strategy of attacker dynamically. We can see that, for type-I attack, honeynet learns in 5 iterations. To distinguish between type II and III, honeynet takes 15 slots. When the attacker deviates from type I, honeynet learns it on the next iteration. However, when the attacker is in type II and changes its strategy, honeynet takes 2 iterations to detect the change in strategy of attack.

Figure 4.5b depicts a scenario where the attack strategy is of type II. It changes its targeted SU transmission characteristics dynamically. For the first phase, attacker aims characteristics 1 and then 2 and then 3. We can see that for imperfect

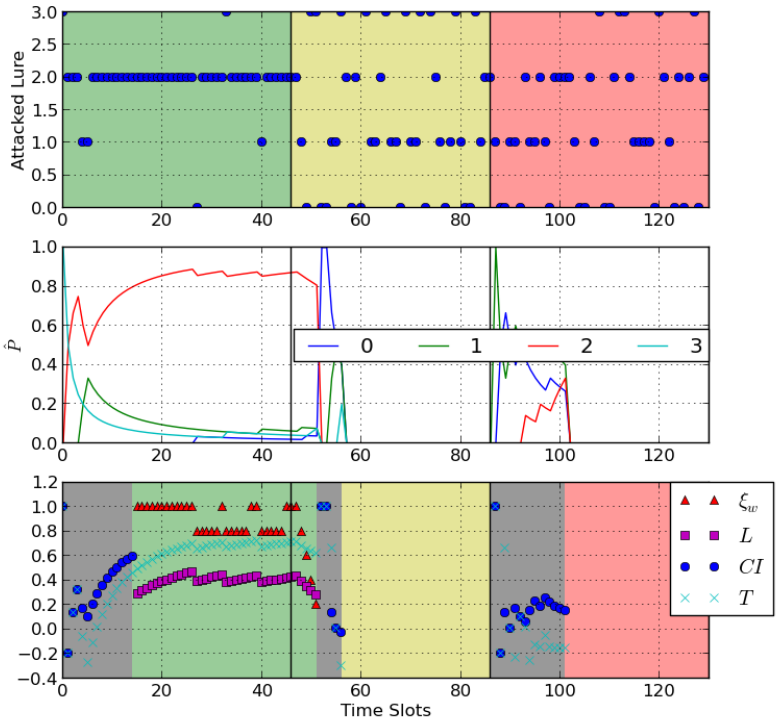


FIGURE 4.6: Honeynet learning phase corresponding to attacker's strategy change from II to I and then to III

scanning, the attack may actually happen on a different lure. Honeynet identifies the correct strategy and particular type of attack in 15 iterations. We can see that for this particular simulation, honeynet detect attack strategy change after 3 iteration in the first case and after 2 iterations in the second one.

Figure 4.6 depicts a simulation scenario where attacker changes its type from II to I and then to III. Here, we can see that honeynet takes 5 iterations to learn that attacker has changed its strategy from type II to type I. From all 3 plots we can see that both CI and T gives indication of learning efficiency.

When honeynode is placed with the wrong lure, legitimate communications are disrupted. We see honeynet detects the regime change very quickly, which decrease the loss. Mainly, the loss is during the learning phase when CR-Honeynet does not deploy honeynode. To see how long does it take for the honeynet to detect

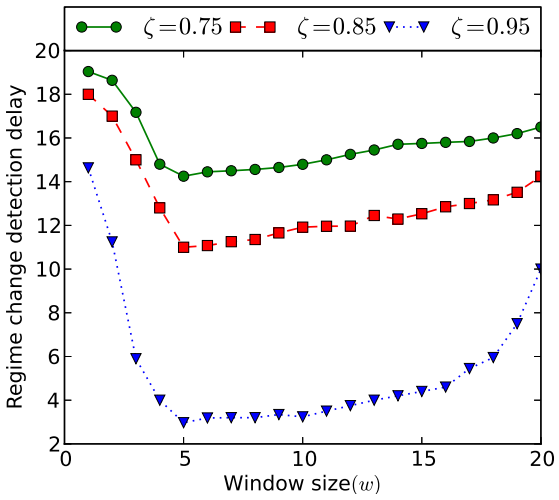


FIGURE 4.7: Regime change detection delay for type II attacker

the regime change while in type II lure strategy, we present a comparison to select optimal window size in Figure 4.7. An attacker of type-II strategy is attacking a particular lure with probability ζ. We simulated for 3 different values of ζ. For every value of ζ and w, the simulation is run for 100,000 slots to ensure accurate results. In this simulation, the attacker is changing its targeted transmission characteristics randomly with mean interval of 100 steps. We can clearly see that, window size $w = 5$ provide optimal result i.e. it can detect regime change very quickly and efficiently.

4.3.4 Overall system performance

We now code an *Event Driven Simulator* to compare the system performance between using honeynet and not using honeynet for an infinite buffer CRN. For simplicity, we have considered 20 SUs and kept $\xi = 0.8$. We vary average packet inter-arrival time (λ) to examine system performance with varying load. We observe that for all values of λ, with CR-Honeynet the average packet dropping probability is 0.01, while without honeynet results packet dropping probability of 0.05. Figure 4.8 provides the comparison of average queuing delay for a SU. From the figure, we can conclude that, using honeynet for lower λ is highly beneficial as

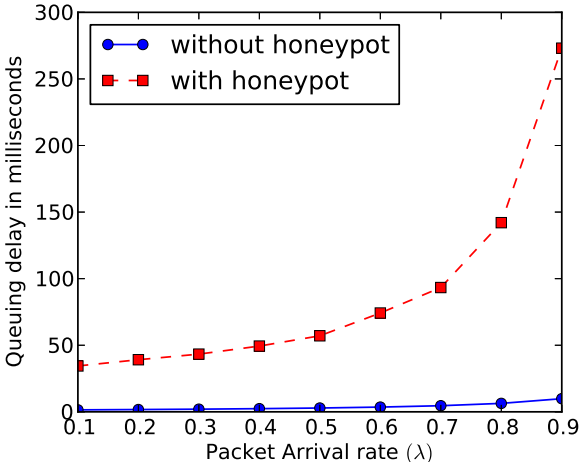


FIGURE 4.8: Average queuing delay for $\mathcal{N} = 20, \xi = 0.8$

packet drop is minimized. Better packet delivery ratio is achieved at the cost of higher packet delay. In our future work we shall try to get an estimation of ξ^* that can regulate CRN to use honeynode or not, depending λ and traffic type (elastic, non-elastic, real-time etc.)

4.4 System development

4.4.1 Testbed setup

We designed the testbed with multiple transceiver, a central controller which would get data from all the transmitters, and an intelligent attacker. The system was built using USRPRadios, each connected with one laptop. The spectrum usage is visualized using a spectrum analyzer. This setup uses one transmitter for each channel in the central controller with one of the transmitters being the honeypot. The central controller has a jamming detector and the transmitter has a controller that would allow it to change frequencies while running. Two different attackers were designed as well. One intelligent attacker that would listen in the the transmissions on the network and attack one based on a target characteristic.

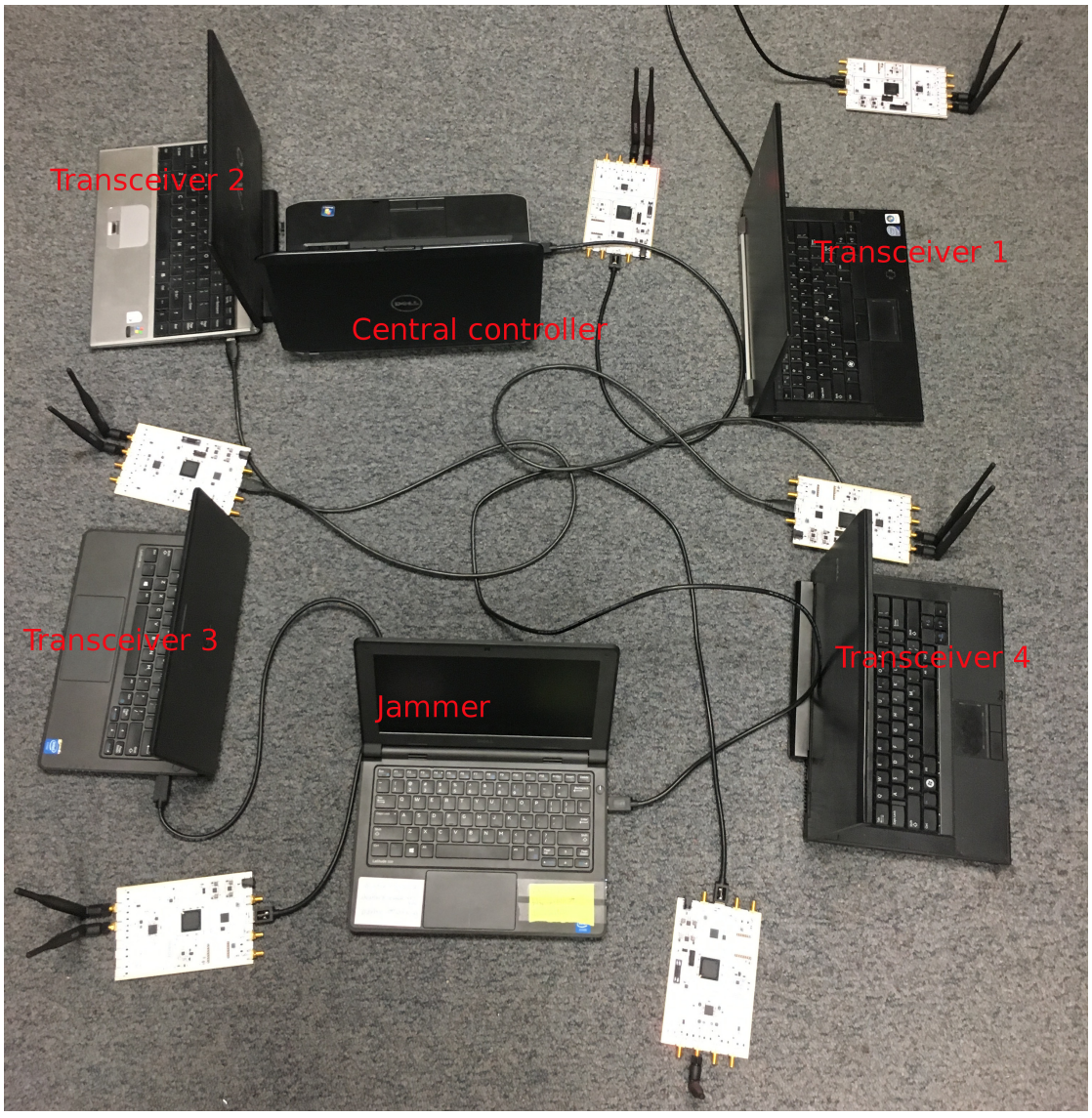


FIGURE 4.9: Testbed Setup

The other was much more simple and was manually moved between us to attack specific channels that we would choose. Figure 4.9 illustrates of our testbed. Next we will discuss how each component worked, what it does, how it does it, and finally discuss the testing.

4.4.2 Central controller design

The central controller (CC) in this system was designed first to work with two channels and then it was updated to work with four. In this setup, one of the

transmitters is designated as the honeypot. CC monitors the incoming data and the signal strengths for each channel separately. If a anomaly is detected for one channel (such as data is not decoded while signal strength is above receive threshold), jamming flag is raised. If CC detects a channel other than the honeypot then it trigger channel switching where the receiver will swap frequencies between the jammed channel and the honeypot so that the jammed channel will now be back on an unjammed frequency. Through a backup common control channel, the message is relayed to the transmitter so that they can use proper frequencies. If the swapped channel is still not receiving data for up to ten seconds, then the receiver will assume the channel is no longer transmitting and swap it with the honeypot again and leave that channel alone until it starts receiving data from it. Swapping back and forth like this also makes the system a little more interference resistant if interference between channels is not allowing data to get through on a channel. Now we shall describe the design of CC.

4.4.2.1 Blocks

For the initial two transmitter receiver, the blocks used were a USRP Source, a Message Strobe, a Jamming Detection/Defense, and two each of the following: Frequency Xlating FIR Filters, GFSK Demods, Packet Decoders, File Sinks, Byte Sensors, and UDP Sinks. For the updated four transmitter receiver, the blocks used were a USRP Source, a Message Strobe, a Jamming Detection/Defense and four each of the following: Frequency Xlating FIR Filters, GFSK Demods, Packet Decoders, File Sinks, Byte Sensors, and UDP Sinks.

4.4.2.2 Parameters

There are seven parameters being used in the CC flowgraph as can be seen in Table 4.1. The center frequency designates which frequency for the USRP source

Table 4.1: Channel parameters

Parameter	**Variable Name**	**Default Value**
Center Frequency	freq	2.44GHz
Number of Channels	num_c	4
Channel Bandwidth	c_width	200KHz
Band Pass Filter Low Cutoff	low_cutoff	-70KHz
Band Pass Filter High Cutoff	high_cutoff	70KHz
Sample Rate	samp_rate	800KHz
Guard Band	guard_band	30KHz

to listen to. The number of channels is used in multiple calculations including to determine the overall sampling rate of the CC. The channel bandwidth is set to designate how wide each channel band and is used to calculate the overall sampling rate as well as the center frequency of each channel. The low and high band pass filter cutoffs are used by the Frequency Xlating FIR Filters to provide guard bands for the channels. The Sample Rate is the overall sampling rate of the entire flowgraph and is set by the number of channels multiplied by the channel width. The guard band is used to help determine the high and low band pass filter cutoffs.

4.4.2.3 Flowgraph

The The data is received from the USRP Source as a wide band transmission that is then filtered and shifted by the Frequency Xlating FIR Filters for each channel in the network. This shift and filter isolates a single channel's data so that it can pass through GFSK Demodulation. Once demodulation is complete, the data gets passed to a Packet Decoder. At this point there is no more processing to be done to the data so it gets sent to the Byte Sensor and the File Sink. When the Byte Sensor receives bytes it uses the GNURadio message passing system to let the Jamming Detect/Defense block know how many bytes it received and at what time it received them. As long as the Byte sensors for the channels are receiving bytes, the Jamming Detector/Defense block doesn't do anything. If it stops receiving

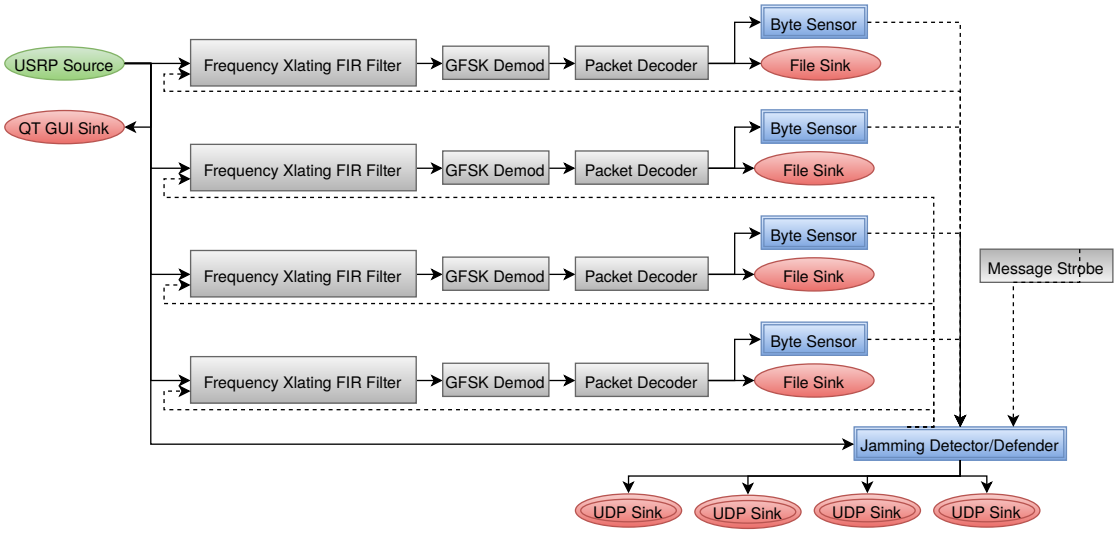


FIGURE 4.10: Flowgraph of the central controller designed in GNURadio

bytes for a certain amount of time, it swaps the honeypot's frequency with the jammed channel's frequency. If there are still no bytes received for that channel on the new frequency the defender assumes it is off. The UDP Sinks are used as a control channel between the receiver and transmitters . The actual flowgraph for a four channel receiver can be seen in Figure 4.10.

4.4.2.4 Custom blocks

The two custom blocks that we built for the receiver are the Byte Sensing block and the Jamming Detection/Defense block. The Byte Sensing block is very simple in design. It only had one input which accepted bytes from the packet decoder and one message output that connected to the Jamming Detection/Defense. Since the work function is only called when the block receives data, the block will only work when the Packet Decoder actually decodes bytes of data. When it does receive bytes, the block uses GNURadio's message passing system to let the Defender know what time that channel received bytes and which channel this sensor is connect to. We also had to build the Jamming Detector/Defender. This block has a message input for each transmitter connected to the block which was two and four in testing, one 64 bit complex input taken directly from the USRP source,

a message output for each channel's Frequency Xlating Fir Filter connected to the receive which was again two and four in testing, and a 32 bit floating point output for each channel's UDP sink. The block contains many different variables to store data about each transmitter including the time of the last known byte, whether the channel is jammed, which frequency each transmitter is on, and many more. The block loops through each transmitter in its list first checking to see if the transmitter it is currently looking at is the honeypot. If it is then just ignore it and move on to the next transmitter, if it is not the honeypot then it checks how long it has been since the it has received a byte from that channel. If it has been at least three seconds since the last byte it received, the defender assumes the channel is jammed and swaps it's frequency with that of the honeypot in an attempt to unjam the transmitter. Then if that receiver has still not received any bytes from that transmitter after 10 seconds, the receiver assumes it is off and swaps it with the honeypot again and will not check on that transmitter again until it starts to receive bytes from it again. For the defender to swap frequencies, it creates a message with the new frequency and send it to the Frequency Xlating Fir Filter for that channel and also sends the new frequency through the floating point output of the corresponding transmitter so it also knows to change. This ensures that not only does the frequency change but the being sent to the receiver is still going to the same place for each transmitter.

4.4.2.5 Limitations

The most profound limitiation in the receiver is the interference that is sometimes created by the transmitters as they move between frequencies. This interference can sometimes cause a transmitter to stay jammed even after it moves. A temporary solution has been found in the form of adding gain control to the transmitters and the attacker and also by having the frequency of the transmitter swap with

the honeypot again if it persists. This can sometimes put the transmitter back into a favorable position where it can get its data through.

4.4.3 Transmitter design

The transmitter for this system was designed to load a file, encode, modulate, and filter the data, and then send the data through the USRP. It is also built to be able to change frequencies during runtime using the control channel created with the UDP blocks. This gives the receiver the ability to monitor the data coming from each individual transmitter and if data is not coming through, the receiver can tell the transmitter to change frequencies.

4.4.3.1 Blocks

Each transmitter includes a UDP Source, File Source, Transmitter Controller, Packet Encoder, Band Pass Filter, Multiply Constant, QT GUI Sink, USRP Sink, and QT GUI Range.

4.4.3.2 Parameters

There are five parameters being used by the transmitters as can be seen in Table 4.2. Center frequency is different for each transmitter and is the frequency that the USRP is transmitting on. The sample rate is the bandwidth of the channel. The guard band determines the high and low cutoffs for the band pass filter. The gain is used to set the gain of the USRP which can actually be controlled by the user at runtime for testing purposes. The payload size is used in the UDP source to determine the size of the UDP packets being sent by the receiver.

TABLE 4.2: Transmitter parameters

Parameter	Variable Name	Default Value
Center Frequency	freq	2.44GH
Sample Rate	samp_rate	800KH
Guard Band	guard_band	30KH
Gain	gain	Varies by Transmitter
Payload Size	payload	1024

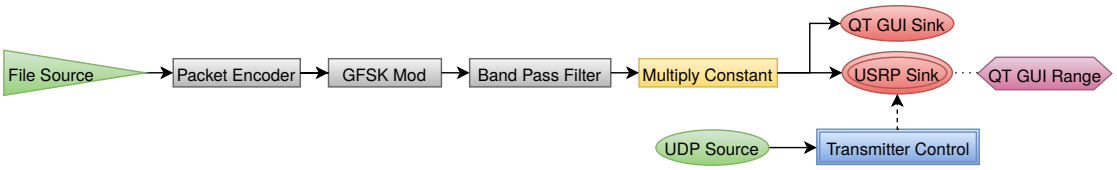


FIGURE 4.11: Flowgraph of the transmitter designed in GNURadio

4.4.3.3 Flowgraph

Figure 4.11 shows the flowgraph of a transmitter. The data being transmitted starts as a file being loaded in by the file source block. When data is loaded it is sent to the packet encoder where it is encoded for GFSK modulation. After that the data goes through a band pass filter to get rid of any undesired frequencies in the signal. Next the data passes through a multiply block which is used to help fine tune the signal. Once done it is sent through the USRP for transmission. Each transmitter also contains a UDP source block which is connected to the UDP sink blocks located within the receiver. This data is sent to a transmitter control block we built to change to a new frequency determined by the receiver.

4.4.3.4 Custom blocks

The Tranmitter Control block was designed to take data from a UDP Source. This data is the new frequency for the transmitter to switch to if the receiver detects jamming. When it receives this data, it uses the GNURadio message passing system to send a message to the USRP to set the new frequency.

4.4.3.5 Limitations

The limitations of this transmitter is changing frequencies depends on the connection between the UDP source and sink blocks in the transmitter and receiver retaining their connection. Also when the transmitter is changing frequency sometimes its data will go into a different transmitters file sink because it didn't switch fast enough.

4.4.4 Attacker Design - Intelligent

For this system, we had to design an attacker with the capability of sensing the network and then attacking channels based on a target characteristic. The two most prominent characteristics we tested against were high transmission power and longest transmission time. In mode 1 the attacker would sense the network and after a specific amount of time it would attack the channel with the highest transmission power. In mode two, after time spent sensing, the attacker would attack the channel that had been transmitting the longest during the sensing time.

4.4.4.1 Blocks

To build the intelligent attacker we had to use a USRP Source, Stream to Vector, FFT, Vector To Stream, Jammer, and USRP Sink.

4.4.4.2 Parameters

There were five parameters used by the intelligent attacker as can be seen in Table 3. The center frequency sets the frequency that the USRP Source is listening to but it does not set the frequency that the USRP Sink is set to initially. This is because the USRP Source will always listen to 2.44e9GHz whereas the USRP

TABLE 4.3: Intelligent attacker parameters

Parameter	Variable Name	Default Value
Center Frequency	freq	2.44GH
Number of Channels	num_c	4
Channel Bandwidth	c_width	200KH
Sample Rate	samp_rate	800KH
Guard Band	guard_band	30KH

FIGURE 4.12: Flowgraph of the intelligent attacker with sensing capabilities

Sink's frequency will change each time the attacker targets a new channel. The number of channels is set so that the jammer knows how many channels it needs to account for in the FFT data. The channel bandwidth is set so that the jammer knows how wide the band is for the channels. The Sampling rate is the overall sampling rate of the network. The guard band is used to set the high and low cutoffs for the bandpass filter. This is needed because we are testing the system with the attacker only being able to attack a single channel at a time.

4.4.4.3 Flowgraph

This attacker takes data from a USRP Source that is set to the same frequency as the Receiver's network. This data is output as a stream which is then turned to a vector of size 1024 by the stream to vector block. This newly created vector is passed to an FFT block which performs the transformation and then passes the data to the Jammer/Sensor. After analysing the data for a set amount of time, the Jammer will then beginning outputting a jamming signal on a specific channel based on the attack mode of the jammer. This jamming signal is put through a band pass filter and then sent to the USRP Sink where it is broadcast on the network. The flowgraph for this attacker can be seen in Figure 4.12.

4.4.4.4 Custom blocks

Only the jammer/sensor block was created for the intelligent attacker. This block has one input that accepts 64 bit complex vectors of size 1024. It also has an output of a stream of 64 bit complex numbers and a message output that is used to change the frequency of the USRP. The block starts by taking in a vector which is the FFT data of the network if it is in sensing mode. Then it splits the data based on the number of channels present in the network. Each segment represents the data on a specific channel. This the block loops through the segments and checks the average of that segment against a threshold. If the average is greater than that threshold then the block will update the information it has for that specific channel. Namely the strength of the transmission and how long it has been on. If the block is in jamming mode then it will use the data it gathered from sensing and choose the channel which matches it's search criteria. Then it outputs the jamming signal on that channels frequency. The block will jam that channel for a set amount of time before it goes back into sensing mode and it repeats the process. In mode 1 the intelligent attacker will target the transmitter with the highest transmission power and can be seen in Figure 4.13. In mode 2 the intelligent attacker will target the transmitter that has the longest transmission time as can be seen in Figure 4.14.

4.4.4.5 Limitations

One of the only limitations of this attacker is noise in the environment can give a false positive that the channel is on. Although this isn't much of an issue because the noise only registers for a very short amount of time and the power is usually really low. This means it usually wont change much in regard to the Jammer's sensing time.

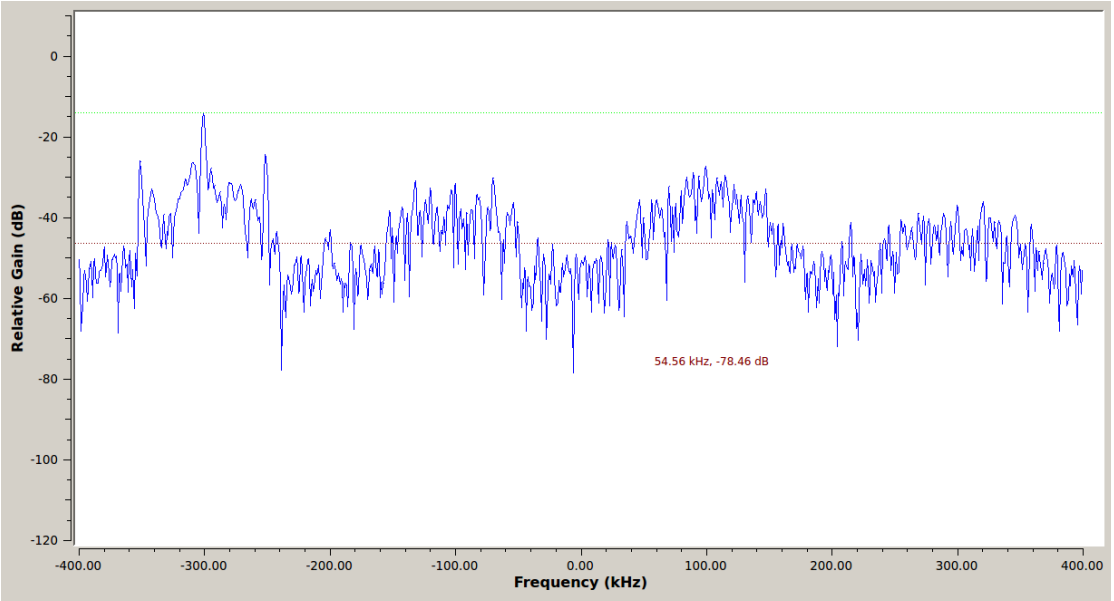


FIGURE 4.13: If intelligent attacker is in mode 1, the transmitter in channel 0 (-300KHz) will be targeted because it has the highest transmission power at that time.

4.4.5 Attacker design - manual

The manual attacker was designed mostly as a testing and debugging tool. It enabled us to test different scenarios while initially designing the system and it also retained its usefulness after the system was built to test specific jamming patterns.

4.4.5.1 Blocks

The manual attacker had by far the simplest flowgraph only using a Signal Source, Band Pass Filter, USRP Sink, QT GUI Sink, and two QT GUI Ranges.

4.4.5.2 Parameters

There were only three parameters used by the manual attacker as can be seen in Table 4.4. The center frequency is the frequency that the USRP is transmitting on and can be changed at runtime by the user to transmit and jam specific frequencies.

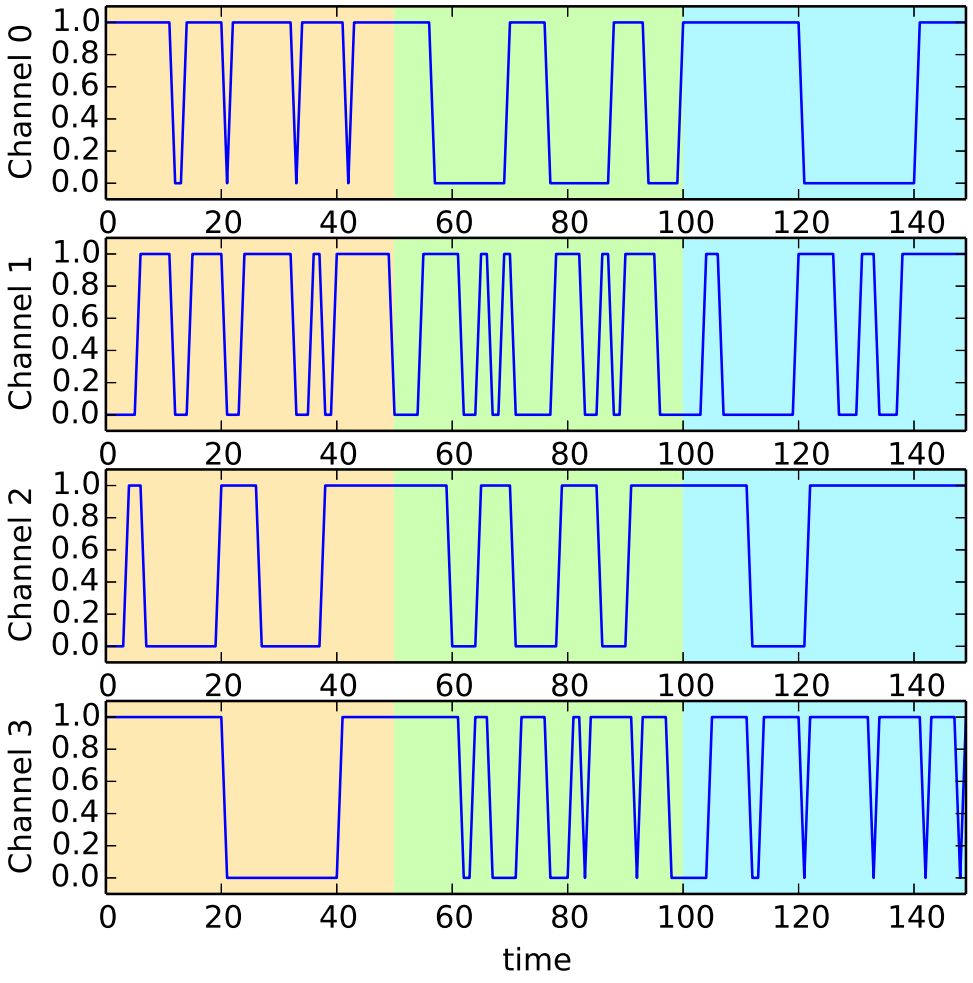


FIGURE 4.14: If intelligent attacker is in mode 2, in the first time slot channel 0 will be targeted, in the second time slot channel 3 will be targeted, and in the third time slot channel 2 will be targeted.

The sample rate is the band width of the channel. The guard band is used to determine the high and low frequency cutoffs for the band pass filter. The gain is used to set the transmission power of the USRP and can also be changed at runtime to change the strength of the jamming.

TABLE 4.4: Manual attacker parameters

Parameter	Variable Name	Default Value
Center Frequency	freq	2.4397GH
Sample Rate	samp_rate	800KH
Guard Band	guard_band	30KH
Gain	gain	Varies by Transmitter

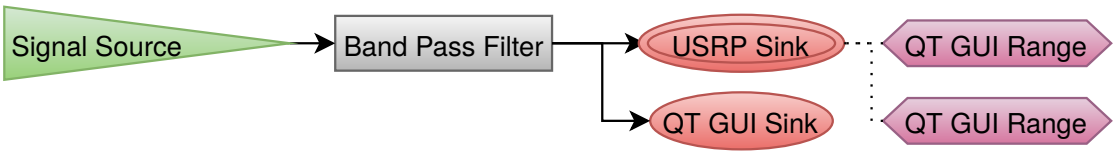


FIGURE 4.15: Flowgraph of the manual attacker that can be changed at runtime to attack a specific channel

4.4.5.3 Flowgraph

The flowgraph is depicted in Figure 4.15. It starts with a signal source that outputs a cosine wave. That signal is then sent to a band pass filter to remove any undesired frequencies in the signal. After filtering the signal, it is sent to the USRP for transmission. While running the user is able to change the gain and center frequency of the USRP to jam specific channels are different strengths.

4.4.5.4 Custom blocks

No custom blocks had to be built for this design. It was done entirely using the built-in blocks of GNURadio.

4.4.5.5 Limitations

The only limitation of the manual attacker is that it must be manually moved to attack different channels. On its own, it contains no sensing or detection capability. This attacker represents one that just chooses a channel to attack and then jams it. No sensing is done to see which channel would be the best one to attack but it also can't really be lured to attack a specific channel.

4.4.5.6 Experiments

Two Transmitter Receiver : experimenting with this system began with just two transmitters and one receiver. Both transmitters were given a file to transmit but one of them was designated the honeypot. In this set up, when the receiver detected jamming on the none honeypot transmitter it would swap the frequencies without a problem and the data would continue to come through. We test this with both the manual attacker as well as the intelligent attacker and in both cases when the receiver detected jamming the channels would swap. The only issue encountered with this was again the interference between the two transmitters. Although most of the time it would not be much of an issue, sometimes it would cause no data to come through the non-honeypot transmitter at all making the receiver think it was no longer transmitting. This stage was just for testing the initial design and since it was working, we moved on to the next part.

Four Transmitter Receiver : once we got the system working for two transmitters, we moved on to four transmitters. In this case three of them would be actively transmitting and the fourth would be the honeypot. Interference was a much bigger issue in this experiment because there were now two extra transmitters to deal with. To help minimize interference, we increased the size of the guard bands and updated the defender to swap the jammed transmitter with the honeypot again after 10 seconds of no received bytes. This extra swap would help get some extra bytes from that transmitter while each one tried to stabilize in its new frequency with its new neighbors. With the issue of interference partially improved, we could begin testing the system with the attackers. A screen shot of the receiver running with four transmitters on can be seen in Figure 4.16.

When testing with the manual attacker, the defender would place the honeypot where the attacker was and only move if some interference made another channel seem like it was being jammed. As the manual attacker was moved to another

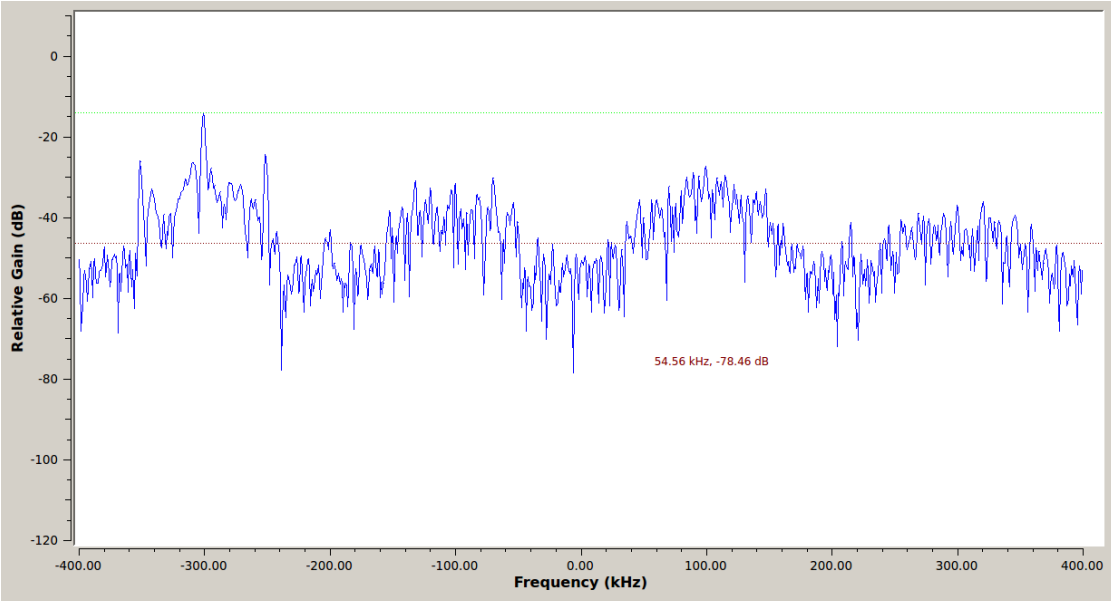


FIGURE 4.16: Screenshot of the receiver with four active transmitters

frequency, the honeypot would follow it. After numerous tests with the manual attacker we then moved on to testing with the intelligent attacker. When the intelligent attacker was introduced to the system, it would target either the transmitter with the highest transmission power or the longest transmission time. When the defender detected the presence of jamming on one of those channels, it would move the honeypot to the jammed frequency until it detected another being jammed.

4.4.5.7 Experiment results

In the testbed, we have deployed an intelligent attacker (i.e. type-II). We have set four sets of characteristics based on transmission power, packet arrival rate, packet length and packet inter arrival gap. Now, the attacker chooses a target characteristics and scan through the channel to detect the target channel. It also changes its strategy dynamically. Figure 4.5b depicts the experiment results. The attacker chose the lure 1 and keep attaching on that characteristics. The defender places a honeynode at the interval 15. The attacker changes its strategy at interval 35 and again in 70. The honeynet's learning period is colored gray. The green color means the defender is using active decoy. The results can be compared with

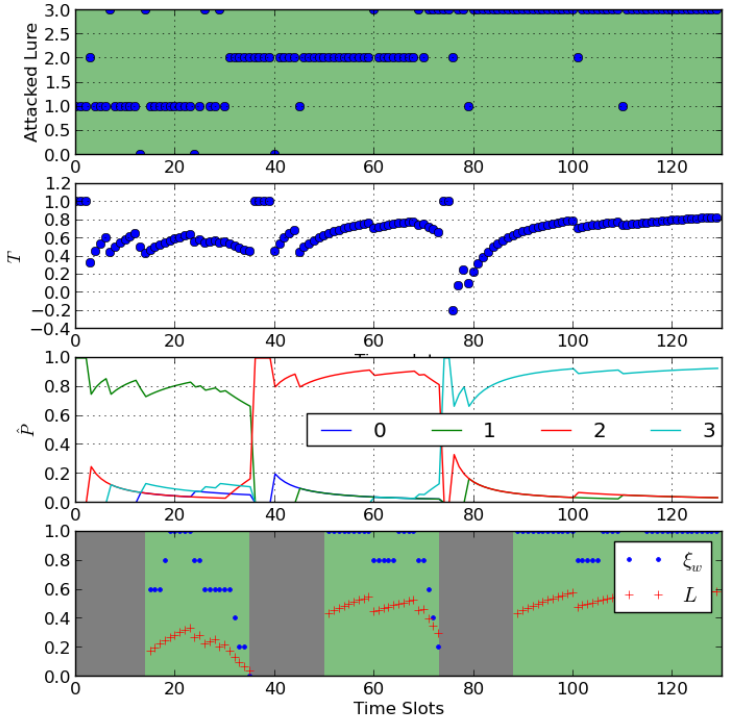


FIGURE 4.17: Experiment result with 4 lures characteristics

the results of the simulation described earlier and clearly support the effectiveness of the CR-Honeynet.

4.5 Queuing behavior of CR-Honeynet

4.5.1 Queuing characteristics

In earlier telecommunication networks, voice packets were generated at fixed rates or at fixed burst sizes [100–104]. For this kind of system the inter-arrival time is fixed and the value depends on the codec (voice digitization technique) used [100, 101]. Voice activity detection and Silence suppression techniques introduces randomness in packet arrival time.

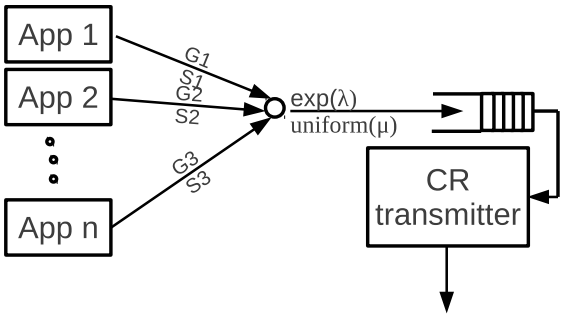


FIGURE 4.18: Depiction of how packets arrive to queue

With the increase in usage of multimedia applications on smart-phones the nature of the traffic flow is very complex to model. Because of the independence between sources, a memoryless inter-arrival time may be a good model. This observation is supported by statistical analysis. The studies carried out in various experiments [105–110] have concluded that when many different applications are merged, the packet arrival process tends to follow Poisson process. Figure 4.18 provides a depiction of how packets from different applications flow to a queue. We use λ_i to denote the rate of the Poisson process of packet arrivals at SU labeled i, and $\{N_i(t); t \geq 0\}$ to denote the corresponding arrival process. When a single queue is analyzed, we drop the subindex i.

Each SU is modeled as a FCFS (first come-first served) queue with one server. Packets arrive according to a marked Poisson process with rate λ and "marks" specifying the packet size in number of bytes. In our model, the marks $\{Y_1, Y_2, \ldots\}$ are independent and identically distributed uniform random variables. The aggregate byte arrival rate is thus $\lambda\,\mathbb{E}(Y)$. Each SU can transmit at a fixed data transmission rate. Therefore, the service time of a packet of size Y_n is, $S_n = (Y_n/data\ rate)$ and it has uniform distribution $U(\ell_1, \ell_2)$, with mean $\mathbb{E}(S) = (\ell_1 + \ell_2)/2$ and maximal service rate, $\mu = 1/\mathbb{E}(S)$.

During the transmission periods of length T_t, the model corresponds to a $M/G/1$ queue [94] where the service time of consecutive packets $\{S_n\}$ are independent and identically distributed. During the sensing periods of length T_s and transmission

periods when the SU is chosen as a honeynode our server stops servicing the queue, which nonetheless continues to accumulate arriving packets. The effect of an attack during a transmission period when the SU is not a honeynode is that all packets transmitted in that slot are lost.

The two performance criteria of interest are the (stationary) average waiting time in queue per packet (W_q), and the average packet drop rate (PDR). In the case of infinite buffer PDR_i is also the long term probability that the i-th SU is attacked, that we call θ_i.

4.5.1.1 Queuing model with vacations

For simplicity, we assume that there are more free channels than the number of SUs in the CRN. In this section we assume that an SU is chosen to be "sacrificed" as a honeynode at every transmission period. If an SU is chosen as a honeynode, then all the new arriving packets join the queue and wait until the next transmission period, where the SU is not chosen as a honeynode. The analysis of this section assumes a random policy, where the i-th SU is chosen as honeynode with probability p_i, independently of past assignments and of the state of the CRN. Other benchmark policies (such as round robin) will be discussed in later sections and compared via simulation experiments.

The amount of service time that must be postponed at the start of a sensing period is either 0 (when the server is idle at time of sensing) or it has the value of the random variable $\tilde{S}$ representing the fraction of service time that must be postponed. In steady state, if $\rho = \mathbb{P}(\text{the server is busy})$ then the server is not idle with probability ρ. Thus, calling X the fraction of service that must be postponed, we have:

$$X = \begin{cases} 0 & \text{w.p. } 1-\rho \\ \tilde{S} & \text{w.p. } \rho \end{cases}$$

We now characterize the random variable $\tilde{S}$, Condition on the event that the sensing period starts when the queue is still not empty. When transmission starts, consecutive service times $S_1, S_2, \ldots$ accumulate until the last service that does not fit into transmission. We now provide precise definitions and results. Let $M(t) = \min(n\colon S_1 + \ldots + S_n \leq t)$. This is a renewal process and it indicates the times of start of successive service epochs. Call

$$J_n = \sum_{j=1}^{n} S_j,$$

then for time $t = T_t$ we are interested in what is known as the "age" or "backward recurrence time" of the renewal process $M(t)$ at time $t = T_t$:

$$\tilde{S} = T_t - J_{N(T_t)}.$$

For a renewal process with no preemption, the distribution of this variable and its expectation can be calculated asymptotically [94]. In our model, where "many" services can be completed during transmission time (specifically, when $\ell_2 << T_t$) we can argue that $\tilde{S}$ will have this known asymptotic distribution as an approximate distribution, so that

$$\mathbb{P}(\tilde{S} \leq x) = \frac{1}{\mathbb{E}(S)} \int_0^x (1 - F(u))\, du$$

where $F(u)$ is the distribution corresponding to the uniform random variable S between ℓ_1 and ℓ_2.

Lemma 4.1. *Assume that* $\mathbb{P}(\tilde{S} \leq x) = \frac{1}{\mathbb{E}(S)} \int_0^x \mathbb{P}(S > u)\, du$. *Then*

$$\mathbb{E}(\tilde{S}) = \frac{\mathbb{E}(S^2)}{2\mathbb{E}(S)}; \quad \mathbb{E}(\tilde{S}^2) = \frac{\mathbb{E}(S^3)}{3\mathbb{E}(S)}.$$

Proof: Let g be any differentiable function with bounded derivative. Call $f(\cdot)$ the density of the service time S. Using calculus it is straightforward to calculate:

$$\int_0^\infty g'(x)\,\mathbb{P}(S > x)\,dx = \int_0^\infty g'(x)\int_x^\infty f(y)\,dy = \int_0^\infty dy\left(\int_0^y g'(x)\,dx\right)f(y)$$
$$= \int_0^\infty g(y)f(y) - g(0) = \mathbb{E}(g(S)) - g(0).$$

Thus, using $g(x) = x^2/(2\mathbb{E}(S))$ we obtain the result for $\mathbb{E}(\tilde{S})$ and using $g(x) = x^3/(3\mathbb{E}(S))$ we obtain the result for $\mathbb{E}(\tilde{S}^3)$. □

■

Using this approximation,

$$\mathbb{E}(X) = \frac{\rho\,\mathbb{E}(S^2)}{2\mathbb{E}(S)} \tag{4.6}$$

For any constant $a > 0$,

$$\begin{aligned}\mathbb{E}(a + X^2) &= \rho(\mathbb{E}(a + \tilde{S})^2 + (1-\rho)a^2 \\ &= \rho(a^2 + 2a\,\mathbb{E}(\tilde{S}) + \mathbb{E}(\tilde{S}^2)) + (1-\rho)a^2 \\ &= a^2 + 2a\mathbb{E}(\tilde{S}) + \mathbb{E}(\tilde{S}^2)\end{aligned}$$

which yields:

$$\mathbb{E}(a + X^2) = a^2 + a\frac{\mathbb{E}(S^2)}{\mathbb{E}(S)} + \frac{\mathbb{E}(S^3)}{3\mathbb{E}(S)}. \tag{4.7}$$

We now calculate the *effective utilization factor* for the queue under the random policy. Assuming that the queues are stable, the effective service rate for each of the SUs satisfies the equation:

$$\mu_i' = \mu\left(\frac{T_t - \rho_i\,\mathbb{E}(\tilde{S})}{T_s + T_t}\right)(1 - p_i), \quad \rho_i = \frac{\lambda_i}{\mu_i'},$$

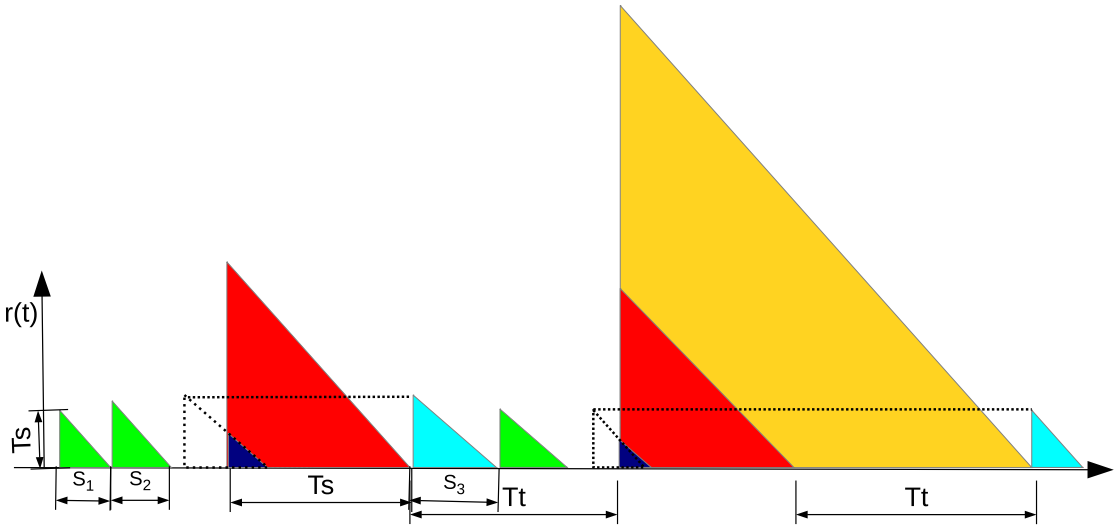


FIGURE 4.19: Path of the residual service of customer currently in service, for an SU. Here, green indicates packet transmission; red is Channel sensing; yellow is Serving as honeynode and blue indicates, the SU postpone current packet transmission as it can not be done within transmission period, cyan indicates postponed packet transmission.

which yields an implicit equation for μ':

$$\mu_i'(T_t + T_s) = \mu \left(\mu T_t - \frac{\mathbb{E}(S^2)\lambda_i}{2\mathbb{E}(S)\,\mu_i'} \right) (1 - p_i) \tag{4.8}$$

Solving the quadratic equation 4.8 gives values of μ_i' that depend on λ_i.

REMARK: If all SUs have equal probability of being chosen (p_i), then the reduced service rate is the same as in the round robin policy.

STATIONARY POLICIES. We now provide an analysis of the stationary queuing delay for the random (or round robin) policies. The analysis is done for each queue, and the subscript i will be dropped from our notation.

Theorem 1. Suppose that i'th SU has incoming rate λ, and that it is chosen as a honeynode independently of the state of the queue, with long term frequency of p. Furthermore, assume that this queue is stable and ergodic and let X satisfy

equations 4.6 and 4.7. Then the stationary delay in queue is:

$$W_q = \frac{R}{1 - \lambda \mathbb{E}(S)(1+\Delta)}, \tag{4.9}$$

where the stationary residual service time is:

$$R = \frac{\lambda\, \mathbb{E}(S^2)}{2} + \frac{\mathbb{E}(T_s + X)^2\,(1-p) + \mathbb{E}(T_s + T_t + X)^2\, p}{2(T_s + T_t)} \tag{4.10}$$

and the correction factor for the vacations is:

$$\Delta = \frac{T_s + \lambda \mathbb{E}(X) + pT_t}{(1-p)(T_t - \mathbb{E}(X))}.$$

Proof: We use the residual service approach [93, 94] to calculate the stationary average delay in queue (assuming that it is well defined) as follows. Sensing periods of length T_s and honeynode periods of length T_t correspond to a "vacation" of the server, and are followed by transmission times of length T_t, during which consecutive packets with varying sizes enter service. Unlike the usual analysis of servers with vacations, here a vacation starts at deterministic times, and not necessarily at the end of busy periods. When not idle, the server can be in three different states: (a) a packet is being transmitted, (b) the server is on vacation, or (c) the current transmission is postponed and the server is waiting for the vacation.

Figure 4.19 shows a typical path of the *residual* time until the completion of the current task (a service, a vacation, or the wait for the vacation), that we call $r(t)$. Under ergodicity, the stationary average residual service is the same as the long term average, given by:

$$R = \lim_{t\to\infty} \frac{1}{t} \int_0^t r(t)\, dt = \lim_{t\to\infty} \frac{1}{t} \left[\sum_{i=1}^{M(t)} \frac{S_i^2}{2} + \sum_{i=1}^{V(t)} \frac{L_i^2}{2} \right] \tag{4.11}$$

where $M(t)$ is the number of arrivals that have entered service up to time t, $V(t)$ is the number of sensing periods up to time t, and L_i is the length of the i-th vacation. It follows that L_i are independent and identically distributed (iid) random variables random variables with composite distribution: with probability $1-p$ the vacation length is $T_s + X$, and with probability p it is $T_s + T_t + X$. For $k \geq 1$, let

$$\tau_k = \min(t > \tau_{k-1} \ : \ Q(t) = 0); \quad \tau_0 \equiv 0$$

be the consecutive moments when the queue empties. The stability assumption implies that the queue empties infinitely often (that is, the state $Q = 0$ is positive recurrent), so that $\tau_k \to +\infty$ with probability one. At these times, $M(\tau_n) = N(\tau_n)$, and $N(t)/t \to \lambda$, because $N(\cdot)$ is a Poisson process. Because the limit R (assuming that it exists) is the same if we consider any divergent subsequence, we can take the limit along the subsequence $\{\tau_k; k \geq 0\}$. In our model $V(t)/t \to (T_s + T_t)^{-1}$. Under ergodicity, long term averages are stationary averages, and

$$\mathbb{E}(L_i^2) = \mathbb{E}(T_s + X)^2(1-p) + \mathbb{E}(T_s + T_t + X)^2\, p,$$

where X satisfies equations 4.6 and 4.7. Applying these results in eq. 4.11 gives expression eq. 4.10.

The rest of the argument is as follows. It is a known property of Poisson processes that sampling a system at Poisson arrival epochs yields states that have a stationary distribution. This is sometimes called " a random snapshot" of the system. In queuing theory this property is also known as "PASTA" (Poisson arrivals see time averages). Using this property an arriving customer will encounter N_q customers in queue, where N_q is a random variable that has the stationary distribution of the queue length. The average wait time is thus the sum of the expected service time of the N_q customers in queue, plus R, plus the contribution of the vacation periods during the waiting time. Call T the required service time for the customers

in queue, then using Little's Law:

$$\mathbb{E}(T) = \mathbb{E}\left(\sum_{k=1}^{N_q} S_k\right) = \mathbb{E}(N_q \mathbb{E}(S)) = \lambda W_q \mathbb{E}(S).$$

Therefore, the stationary delay upon arrival at the queue will satisfy:

$$W_q = \lambda W_q \mathbb{E}(S) + R + v_s(\, T_s + \mathbb{E}(X)) + v_h\, T_t, \tag{4.12}$$

where v_s and v_h are the (expected) number of sensing and honeynode periods (respectively) that fall within the time required to transmit all the N_q customers in front of the new arrival. In the expression above we have used the fact that for every sensing period, the actual vacation time is not just T_s but we must add the lost time from the postponed service (if any). On average, the stationary contribution of this excess is $\mathbb{E}(X)$.

We now proceed to the calculation of v_s and v_h. In order to do so, we will use Wald's theorem [94]. Given T, the actual number of (true) transmission periods required to provide the service for the N_q customers in queue is:

$$\nu_t = \min\left(n : \sum_{i=1}^{n}(T_t - X_i) \geq T\right), \tag{4.13}$$

where X_i is the fraction of postponed service at the i-th sensing period. This is a stopping time adapted to the filtration $\mathfrak{F}_n$ generated by $\{Z_i \equiv T_t - X_i, i \leq n\}$. In addition, Z_n is independent of $\mathfrak{F}_{n-1}$. For our model the random variables $\{Z_n\}$ are bounded, thus absolutely integrable. It is straightforward to verify that $\mathbb{E}(X_n \mathbf{1}_{\{\nu_t < n\}}) = \mathbb{P}(\nu_t < n)\, \mathbb{E}(X)$, and finally, $\mathbb{E}(\nu_t) < \infty$, which follows because $\nu_t \leq T/(T_t - \ell_2)$ w.p.1. Under these conditions, Wald's Theorem ensures that

$$\mathbb{E}\left(\sum_{i=1}^{\nu_t} Z_i\right) = \mathbb{E}(\nu_t)(T_t - \mathbb{E}(X)).$$

Rewrite eq. 4.13 as: $\sum_{i=1}^{\nu_t} Z_i \leq T < \sum_{i=1}^{\nu_t+1} Z_i$ and take expectations to get:

$$\frac{\mathbb{E}(T)}{T_t - \mathbb{E}(X)} - 1 < \mathbb{E}(\nu_t) \leq \frac{\mathbb{E}(T)}{T_t - \mathbb{E}(X)}.$$

In stationary state, we use the approximation $\mathbb{E}(\nu_t) = \lambda W_q \mathbb{E}(S)/(T_t - \mathbb{E}(X))$. In order to calculate v_s and v_h we reason as follows: given the number of honeynode periods, the number of sensing periods is the number of true transmission periods required to exhaust the time T, plus v_h, that is:

$$v_s = \mathbb{E}(\nu_t) + v_h = \mathbb{E}(\nu_t) + pv_s, \implies v_s = \frac{\mathbb{E}(\nu_t)}{1-p}.$$

Replacing now these values in eq. 4.12 and using $\mathbb{E}(T) = \lambda W_q \mathbb{E}(S)$, we obtain

$$W_q = \lambda W_q \mathbb{E}(S) \left(1 + \frac{T_s + \mathbb{E}(X)}{(1-p)(T_t - \mathbb{E}(X))} + p\frac{T_t}{(1-p)(T_t - \mathbb{E}(X))}\right) + R,$$

which yields eq. 4.9, after some simple algebra. ■

4.6 Honeynode selection policies

4.6.1 State dependent policies for uniform traffic distribution

When an SU is chosen as honeynode with initial queue size of Q packets, the queue size at the beginning of the following transmission period is $Q + A$, where $A \sim$ Poisson$(\lambda(T_s+T_t))$. In order to understand the effects of honeynode assignment, we now look at the dynamics of a single channel with an initial queue of a given size. Consider a queue with initial service requirement (in milliseconds): $u = \sum_{i=1}^{Q} S_i$, where $\{S_i; i \geq 0\}$ are iid $\sim U(\ell_1, \ell_2)$. The remaining service time at time t seen

by the server is a stochastic process that follows the dynamics:

$$K(t) = u + \sum_{i=1}^{N(t)} S_i - t, \quad t \leq \tau_u,$$

where $N(t)$ is the Poisson arrival process of packets, with rate λ and $\tau_u = \min(t \leq T_t \ : \ K(t) \leq 0)$ is the time until the queue empties, or until the service stops because a sensing period starts.

This is called the "storage process" and it is dual to the surplus process in the canonical model for risk theory [111]. We are interested in evaluating the probability that the queue empties within the current transmission period, that is, $\mathbb{P}(\tau_u \leq T_t)$. This quantity is known in classical risk theory as the finite horizon "ruin probability". Because there are no closed form solutions, a number of methods have been proposed in the literature to evaluate the ruin probability, mostly when $T_t = \infty$.

In our problem, the packets have an integer number of bytes. If we consider IEEE802.11g channel with data transmission rate of 36 Mbits/sec, the natural time to transmit a single byte, $\delta = \mathcal{O}(10^{-7}$ ms). We consider $u = j\delta$ for $j \in \mathbb{N}$. To discretize the arrival process for small δ, we approximate the Poisson process with an independent Bernoulli trials process with $\mathbb{P}(N(\delta) = 1) = 1 - \mathbb{P}(N(\delta) = 0) = 1 - e^{-\lambda\delta}$. Define the function:

$$\phi_N(j) = \mathbb{P}(\tau_{j\delta} \leq N\delta). \tag{4.14}$$

which defines the probability that the queue will empties within the next transmission period. Then we are interested in solving eq. 4.14 for $N = \lfloor T_t/\delta \rfloor, j = \lfloor u/\delta \rfloor$. First notice that if $j \geq N$ then $\phi_N(j) = 0$, because it takes longer to serve the current packets than the prescribed time horizon. Next, suppose that $j < N$. Conditioning on the event that the first packet arrives during the interval $[(k-1)\delta, k\delta)$,

it is immediate that $\phi_N(j) = 1$ for all $k > j$ (no arrivals while there is transmission, so the queue empties) which happens w.p. $e^{-\lambda j\delta}$. For $k \geq j$ the new arrival has Y bytes, and k bytes have been transmitted. The function ϕ satisfies the recursive equations:

$$\phi_N(j) = e^{-\lambda j\delta} + (1 - e^{-\lambda\delta}) \sum_{k=1}^{j} \mathbb{E}(\phi_{N-k}(j - k + Y))\, e^{-\lambda k\delta},$$

for $Y \sim U(126, 2146)$ is the packet size in bytes. The boundary conditions are:

$$\begin{aligned} \phi_N(0) &= 1 \quad \forall\, N, \\ \phi_0(j) &= 0 \quad \forall\, j, \\ \phi_N(j) &= 0 \quad \forall\, j \geq N. \end{aligned}$$

In principle, the above equations can be pre-calculated starting at $N = 1$ and increasing N, similar to a two-dimensional dynamic programming problem.

At the end of a sensing period, the central controller of the CRN can then evaluate, for every channel $i = 1, \ldots, n$, the probability that it empties if it chosen as a honeynode, using

$$\pi_i = \mathbb{P}(\text{emptying during period}) = e^{-\lambda_i} \sum_{a=0}^{\infty} \Phi_i(u_i + a)\, \frac{\lambda_i^a}{a!},$$

where $\Phi_i(x) = \phi_{\lfloor T_t/\delta \rfloor}(\lfloor x/\delta \rfloor)$ for SU_i.

Notice that if θ_i is the probability that SU i is attacked and p_i is the long term fraction of periods where SU i is chosen as a honeynode, then $\text{PDR}_i = \theta_i((1 - p_i) + p_i(1 - \xi_i))$. In particular, if all channels are equally likely to be attacked then $\theta_i = 1/n$, and if $p_i = 1/n$, then

$$\text{PDR}_i = \frac{1}{n}\left(1 - \frac{\xi}{n}\right). \tag{4.15}$$

This is verified in Section 4.7.3.

Therefore, strategies for honeynode selection may include choosing the SU that has the largest probability of emptying its queue. For a CRN where all SU's have identical traffic (same arrival rates), choosing the SU with highest probability of emptying the queue is equivalent to choosing the SU with lowest queue size. Using, *largest probability of emptying queue* strategy, the CRN with uniform traffic chooses an SU that has the lowest queue at the beginning of a transmission period. For comprehensiveness, we compare this policy with round robin and random honeynode selection. In *random honeynode selection* strategy, one SU is chosen randomly to serve as a honeynode. In *round-robin honeynode selection* strategy, each SU takes a turn to serve as a honeynode in a cyclic order. In Section 4.7.3, we present the system performance of these honeynode selection strategies and also compare the performance of CRN when it does not use a honeynet.

4.6.2 Optimal honeynode selection strategy for non uniform traffic distribution

So far we have considered uniform traffic load among all SUs in a CRN. In this case, state dependent policy of choosing SU with lowest queue size is beneficial in terms of overall system performance as can be seen in Section 4.7.3. However, choosing SU with lowest queue size provides lowest fairness in the case of non-uniform traffic (i.e. SUs with different data rate requirements). It may easily be possible that the SU with lowest traffic is starved of services. This particular SU would be chosen as honeynode most of the time due to lower accumulated packets in the queue compared to other SUs. Repeatedly dedicating one SU over the other SUs results in more queuing delay for this SU. Round robin strategy provides fairness of service but lacks in overall system performance. Section 4.7.6 presents this trade-off.

Queuing delay and PDR define Quality of Service (QoS) measure of different applications. Some applications (such as real time application) can tolerate packet loss but not delay and some applications (such as FTP) can tolerate delay but can not tolerate packet loss. A utility function has to be defined considering the delay and PDR so that CR-Honeynet can determine the best candidate for Honeynode. Utility function varies with the application. In Section 4.7.5, we have used *R-score* as utility function for voice application. We have already stated that packet arrival model is a marked Poisson process where the marks specify the packet size. We can calculate transmission or service time (S_i) for each packet in the queue at the beginning of transmission period. Let's say U_i is the utility function of SU_i that depends on queuing delay and the PDR. U_i^{avg} is the average of U_i observed till the last transmission period. Now we model the honeynode selection strategy to a maximization problem.

$$
\begin{aligned}
maximize: \quad & \sum_{i\in\mathbb{N}} \left(U_i(d_i^{exp}, \text{PDR}_i)\right)^2 \sum_{i\in\mathbb{N}} \psi(i).U_i^{avg} \\
subject\ to: \quad & \exists!\ i \in \mathbb{N}\ (\psi(i) = 1) \\
& d_i^{exp} = \frac{\frac{\left(\sum_{j=1}^{Q(i)} S_j + T_s + T_t.\psi(i)\right)^2}{2T_s + Tt}}{1 - \lambda_i \mathbb{E}_i(S)(1 + \Delta_i)}, \\
& \Delta_i = \frac{T_s + \lambda_i \mathbb{E}(X)}{(T_t - \mathbb{E}(X))}.
\end{aligned}
\tag{4.16}
$$

Where $\psi(i)$ is a indicator function: $\psi(i) = 1$ if SU_i is chosen as honeynode for the next transmission cycle and $\psi(i) = 0$ if it serve as normal SU. Q_i is the number of queued packets in SU_i. $\mathbb{E}_i(S)$ is the expected packet transmission time. PDR is measured from past events.

Determining the SU to select is very easy to calculate from the above mentioned maximization problem. We consider all SUs as possible candidates for honeynode

and plug $\psi(i) = 1$ separately. After calculating the utility we choose the SU that calculates highest according to eq. 4.16.

4.7 Simulation and results

In this section, we first describe our baseline simulation model. After that, we inspect the accuracy of the mathematical model with simulated results. Then, we present the performance of CRN with limited buffer. We build a model that determines when CR-Honeynet should or should not be used. We examine the fairness of honeynode selection strategies with a fairness index and finally, present an optimal honeynode selection strategy that provides better performance with higher fairness.

4.7.1 Simulation parameters and model

We coded a *discrete event simulation* [112], written in Python. in order to analyze CR-honeynet's performance. All arrival rates (λ) are in millisecond domain and mean λ packets per millisecond. As a first step, we consider equal arrival rates $\lambda_i = \lambda$ amongst the SUs. Under this assumption, the randomized and the queue-dependent policies become a randomized policy with equal probabilities, and a minimum queue-size policy, respectively. The data for our model is given in Table 4.5.

In all our simulations, we use the technique of antithetic random variables (ARN) for increased precision. For the infinite queue model, where waiting and loss are monotone functions of the inter-arrival and service variables, ARN ensures variance reduction [112] (we used the inverse function method for generating random variables). For the finite buffer model, because some packets may be lost, it is no longer true that larger inter-arrivals (service times) always have a decreasing

TABLE 4.5: Simulation parameters

Parameter	Symbol	Value
Number of SU	N	20
Packet Service Time	S_n	$\sim U(0.1, 1.7)$ ms
Sensing Period	T_s	50 ms
Transmission Period	T_t	950 ms
Number of attacks / slot		1
Number of honeynodes /slot		20
Number of replication		30
Simulated time		5000000 ms
Warm-up time		100000 ms

(increasing) effect on the delay. Although the theory does not ensure variance reduction for the finite buffer model, we verified this by experimentation.

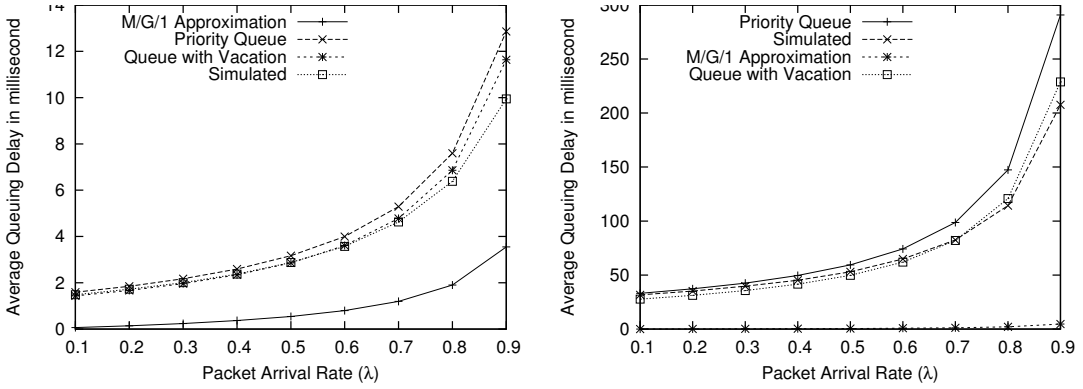
Using a simulated time of $50,000$ time slots entails that the number of packets served in each SU is also a random variable. For each replication of the simulation, we discarded the "warm up" data corresponding to the first 100 time slots. Preliminary simulations were used to choose these numbers, testing for stationarity and a satisfactory precision. For each replication or run of $50,000$ slots we estimated the quantity $(1/N)\sum_i^n W_q(i)$ that we call the average wait time in the queues. We then used 30 independent replications to calculate 95% confidence intervals of the form:

$$\bar{W}_q \pm t_{29,0.975}\sqrt{\frac{\widehat{\mathbb{V}\mathrm{ar}(W_q)}}{30}},$$

where $\widehat{\mathbb{V}\mathrm{ar}(W_q)}$ is the sample variance from the 30 replications. In the plots that follow we do not report these intervals. In a typical simulation with $\lambda = 0.9$ and no honeynode the estimated average wait was 9.849 ± 0.097, which corresponds to a relative error of 1%.

4.7.2 Comparison of approximations

Using parameters in Table 4.5, μ' can be calculated as in eq. 4.8. When $\lambda \in [0.1, 0.9]$ and no honeynodes are assigned ($p_i = 0$) we get the range of values



(A) without Honeynode (B) With one Honeynode

FIGURE 4.20: Average queuing delay for simple cognitive radio network

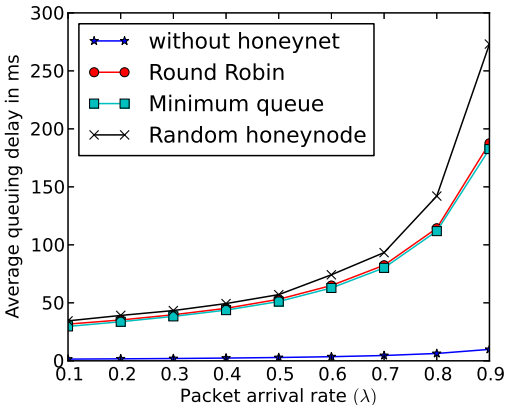
$\mu' \in [1.05513, 1.05551]$. For random honeynode assignment ($p_i = 1/N$), the corresponding range is $\mu' \in [1.00283, 1.00320]$, ensuring stability for all queues. The analytical formulas available hold for the infinite buffer model and are as follows.

M/G/1 QUEUE. To obtain an expression for the stationary average delay or waiting time in the queue, we calculate a first crude approximation using the $M/G/1$ formulas with the effective rates λ and μ' [94].

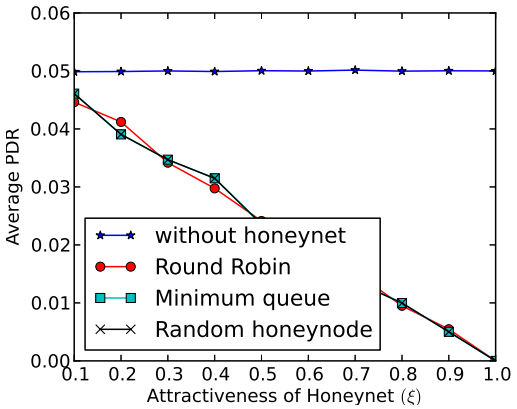
PRIORITY MODEL. A second approximation is based on a $M/G/1$ priority queue [94]. The sensing operation is to be served with higher priority, whereas packet transmission is a low priority job. Formula 4.17 estimates the stationary average queuing delay for a packet with service priority i, when all customer classes arrive according to independent Poisson processes.

$$W_q^i = \frac{\lambda_1 \mathbb{E}[S_1^2] + \cdots + \lambda_n \mathbb{E}[S_n^2]}{2 \prod_{j=i-1}^{i} (1 - \lambda_1 \mathbb{E}[S1] - \cdots - \lambda_j \mathbb{E}[Sj])} \tag{4.17}$$

This formula is only an approximation because the arrival rate of the "sensing" or high priority jobs is $\lambda_1 = (T_s + T_t)^{-1}$, and $S_1 = T_s$ is deterministic. Second high priority job is serving as honeynode where $\lambda_2 = p(T_s + T_t)^{-1}$ and $S_2 = T_t$, while $S_3 \sim U(0.1, 1.7)$ is the original packet service time distribution.



(A) Average Queuing Delay in millisecond for $\xi = 0.8$



(B) Average PDR of all SUs for $\lambda = 0.6$

FIGURE 4.21: Results for CRN with infinite buffer

VACATION MODEL. This corresponds to our formula 4.9. When no honeynodes are assigned, we use $p_i = 0$. For the random honeynode assignment we use $p_i = 1/20$.

We have simulated two scenarios. In the first scenario we assume there is no attacker, hence the CRN does not use honeynet. In the second scenario there is an attacker and the CRN dedicates an SU as honeynode in each transmission period. Figures 4.20a and 4.20b show the results. We can clearly see that average queuing delay is higher for the second case as each SU serves as honeynode at its own slot and delay the packets. This degradation of performance is balanced by achieving lower packet drop while using CR-Honeynet to defend against jamming attack. We have presented these results as benchmark, only to compare the accuracy of our simulation model to other established queue model. The discrepancies are not very visible for smaller values of λ but they become more apparent for heavier traffic regimes, where our vacation formula seems to agree best with the simulated system.

4.7.3 Comparison of honeynode assignment strategies for CRN with infinite buffer

In this section, we have used FTP data transfer where all SUs with uniform load. Figure 4.21a shows the average wait per packet as λ increases with fixed $\xi = 0.8$ and infinite buffer sizes. In infinite buffer systems there is no packet drop for queue overflow, and therefore PDR is independent of λ. The only cause of packet drop is the jamming attack. Simulation results reflect that with $\xi = 0.8$, having no honeynode gives PDR of 0.05 and with one honeynode, PDR is 0.01 for all values of λ.

For the infinite buffer model, if θ_i is the probability that i'th SU is attacked and p_i is the long term fraction of periods where SU_i is chosen as a honeynode (assuming stationarity), then $PDR_i = \theta_i((1-p_i)+p_i(1-\xi_i))$. When $\theta_i = p_i = 1/N$ and $N = 20$ we obtain the linear function $0.05(1 - 0.05\xi)$ as verified in Figure 4.21b. The attractiveness (ξ) does not affect the queue size, which is only dependent on the strategy and the incoming rate λ. Simulation result shows average queuing delay for no honeynode, random , minimum queue and round-robin selection strategies are 3.54 ms, 72.55 ms, 62.1ms and 64.62 ms respectively for all values of ξ. These results clearly say that state dependent policy i.e. selecting honeynode based on minimum queue length is performing better compared with others strategy. Honeynet ensures less packet drop at the cost of increased queuing delay.

4.7.4 Performance of CRN with finite buffer and uniform traffic

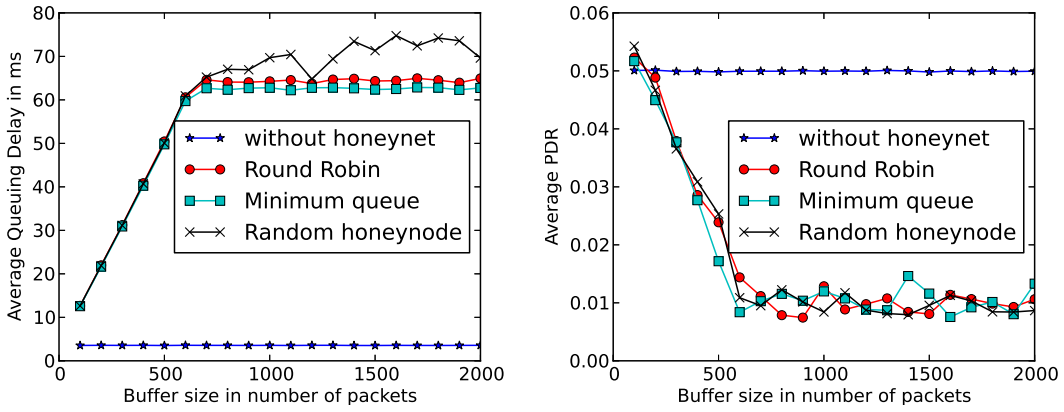
When the queues have limited buffer capacity, incoming packets that can't fit in the buffer are dropped (lost). Even in the absence of attacks $PDR_i \neq 0$. In the absence of analytical models, we use simulations to assess the performance

of various honeynode assignment strategies. Figures 4.22a and 4.22b show the results for the average wait and PDR respectively, as a function of the buffer size, when $\lambda = 0.6$ and $\xi = 0.8$ are fixed. On average $\lambda \times (2T_s + T_t) = 630$ packets would be queued when SU serves as honeynode. There would be queue overflow if buffer is smaller, then SUs have significant PDR due to overflow. For larger buffer, performance is similar to that of an infinite buffer.

We run another set of simulations with buffer size as 400 packets for every SU. Figures 4.23a and 4.23b show the observed average queuing delay and PDR respectively. With low arrival rate λ (below a threshold) honeynode selection strategy based on minimum queue is performing better. This threshold value of λ should be Buffer Size$/(2T_S + T_t) = 0.381$ because one SU can accumulate this amount of packets when serving as honeynode. When λ goes above the threshold, and when the SU is serving as honeynode, it accumulates many packets so that the queue overflows which causes increase in PDR. Below this threshold SUs behave similar to infinite buffer model. These two graphs show a good trade-off between Delay and PDR. With limited buffer and from the two figures it is clear that the system administrator have to come to a conclusion at particular value of λ and ξ and specified buffer size, whether to apply a honeynet or not. At higher value of λ and loss tolerant traffic (such as real time video) not having honeynode as it can not tolerate delay.

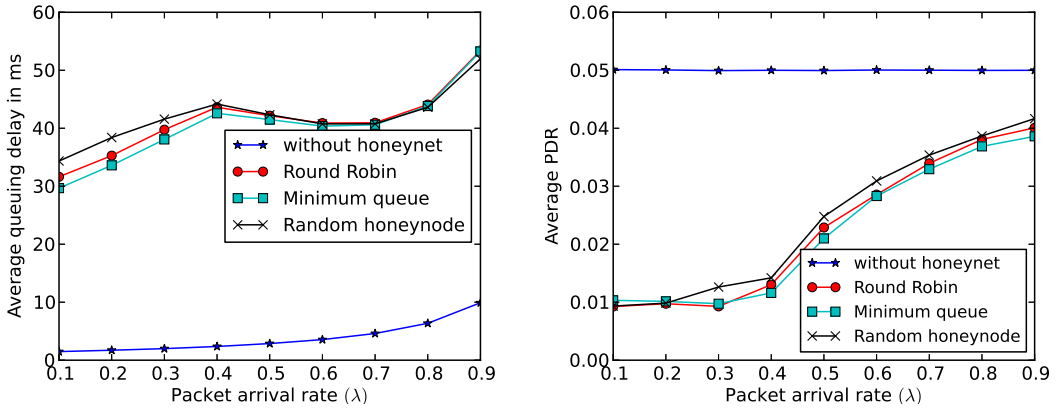
4.7.5 When Honeynet can be applied and when not

We have seen that the choice of dedicating SUs as honeynode comes with a trade-off. We need to analyze when CR-Honeynet is beneficial to use or not. During each transmission period, the CRN assigns an SU as a honeynode. This decreases overall PDR while introducing extra delay to packet transmission. From eq. 4.15, we can see that, for the case of uniform traffic (i.e. all SUs handle equal traffic



(A) Average Queuing Delay in ms (B) Average PDR

FIGURE 4.22: Results varying buffer size of SU with $\lambda = 0.6, \xi = 0.8$



(A) Average Queuing Delay in millisecond (B) Average PDR

FIGURE 4.23: Results for CRN varying λ with $\xi = 0.8$ and Buffer of 400 packets

load), PDR is inversely proportional to the attractiveness of honeynet (ξ). Again, an increase in PDR causes a degradation in the QoS of the system. In this section, we want to determine the threshold, the lowest value of attractiveness for effective honeynet (ξ^*). ξ^* is the point at which the net gain of using honeynet is zero. A CRN achieves a higher performance by assigning honeynode when $\xi > \xi^*$. A honeynet, with effective attractiveness below ξ^*, is not worth of dedicating one SU to serve as honeynode. To determine this threshold, we need to define a performance function of the CRN that takes into account both delay and PDR. *Non real time traffic*, such as FTP can tolerate delay provided all packets are

received successfully. A honeynode with $\xi > 0$ is always beneficial for non real-time traffic because it guarantees lower PDR, compared to not having a honeynode. So, we try to find out ξ^* for *real time traffic* that has a stringent end-to-end packet delay requirement. Higher end-to-end delays degrade the system performance significantly and a packet is considered lost if the end-to-end delay exceeds a certain threshold.

As the first step in analyzing real time traffic, we consider Voice over IP (VoIP) traffic to determine ξ^*. For VoIP services, there are two indicators that define QoS, namely *mean opinion score* (MOS) and *R-score* [3, 4, 113–117]. Both of these indicators depend on end-to-end delay and packet loss. The end-to-end delay or mouth-to-ear delay of voice application is, in turn, composed of three parts, the codec delay (d_{codec}), the playout delay ($d_{playout}$), and the network delay ($d_{network}$). Codec delay and playout delay are dependent on the codec being used and the receiver side buffer respectively. These delays are usually very small (generally between 10 and 50 ms). The network delay is the component that varies with network conditions. Again, the network delay consists of queuing delay (d_{queue}) and transmission delay ($d_{transmission}$). Transmission delay, or the time taken to transmit on packet, is very negligible (0.1 to 1.7 ms). The sensing period, as well as dedicating an SU as a honeynode, introduces a very high queuing delay to the packet transmission. Considering all these factors, end-to-end delay can be written as:

$$d = d_{codec} + d_{playout} + d_{network} + d_{queue} + d_{transmission}$$

Packet loss can happen due to packet drops during transmission and while discarding a packet at the receiver end due to adaptive playout. Packet drop during transmission is the same as the PDR while playout packet loss probability ($e_{playout}$)

TABLE 4.6: Coefficient parameters for calculating loss impairment [3–7]

Codec	Bandwidth (kbps)	γ_1	γ_2	γ_3	Packatization Delay(ms)	Frames/pkt
G.711	64.0	0	30.00	15	1.0	1
G.723.1.B	5.3	19	37.40	5	67.5	1
G.723.1.B	6.3	15	36.59	6	67.5	1
G.729	8.0	10	25.05	13	25.0	1
G.729A+VAD	8.0	11	40.00	10	25.0	2

has to be measured at the receiver end. Total loss probability can be written as :

$$e = \text{PDR} + (1 - \text{PDR})e_{playout}$$

Sengupta *et al.*have defined MOS and R-Score as follows:

$$\text{MOS} = 1 + 0.035R + 7 \times 10^{-6}R(R - 60)(100 - R) \quad (4.18)$$

$$R = 94.2 - (\gamma_1 + \gamma_2 \ln(1 + \gamma_3 e)) - (0.024d + 0.11(d - 177.3)\mathbf{H}(d - 177.3) \quad (4.19)$$

Where, $\mathbf{H}(x)$ is an indicator function. $\mathbf{H}(x) = 0$ if $x < 0$ and 1 otherwise. γ_1, γ_2 and γ_3 are Loss Impairment Parameters that depend on specific codecs that are used for digitization and packetization of voice samples. Coefficient parameters for useful codecs are provided in Table 4.6, which provide the application layer data rate. Every packet that passes through MAC layer has to contain the transmission layer, network layer, and MAC layer headers.

Cole *et al.*[4] have provided a table that signifies the QoS with MOS values which indicates, the higher MOS value, the better the QoS. Again, when MOS is plotted against R-score, it reveals that a system obtains better QoS when R-score is higher. R-score of 80 and above is desired whereas 70 and higher is acceptable. We are taking R-score as the CRN's performance measure and the goal is to maximize the average R-score.

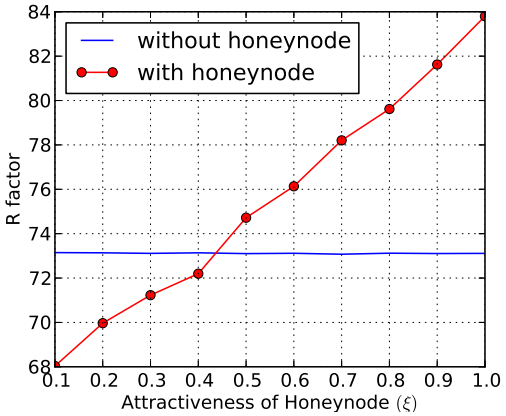
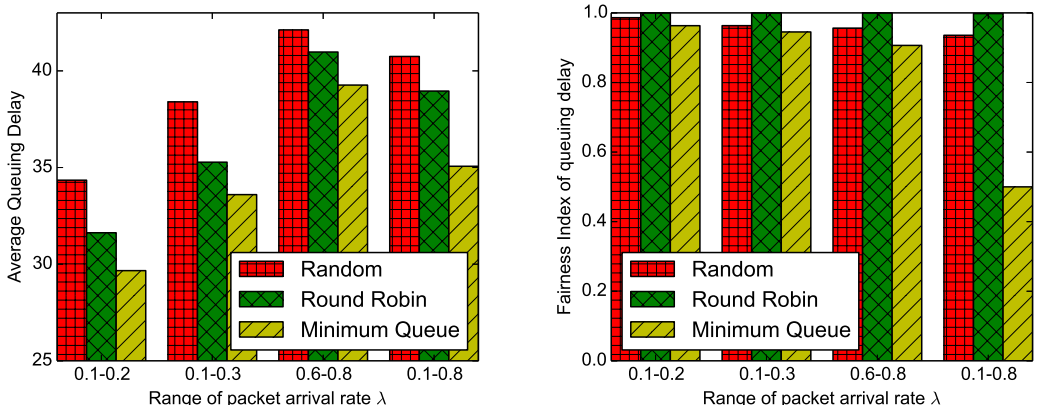


FIGURE 4.24: Average R-score of the CRN

Now, we build another simulation of CRN where all SUs, with infinite buffer, are transferring packets in accordance with G.711 codec [3], for voice digitization. Here, we are considering the minimum queue honeynode selection strategy. Figure 4.24 plots the average R score for the CRN. From eq. 4.9, we see that the queuing delay is irrespective of ξ, where as eq. 4.15 reveals that PDR is inversely proportional to ξ. When all other parameters are unchanged, R-score is proportional to ξ. In other words, an increase in ξ enhances the QoS, which can be seen in the figure. We say, ξ^* is the attractiveness of honeynet for which, $R_{with\ honeynode} = R_{without\ honeynode}$. When $\xi > \xi^*$, the CRN dedicates one SU as a honeynode. When using no honeynode, we observe a R-score of 73.11. We can clearly see here that, for $\xi = 0.44$, the R-score of using honeynet is same as of not using honeynet. For this particular CRN, we can conclude that the lowest effective attractiveness of honeynet $(\xi^*) = 0.44$.

4.7.6 Fairness of performance for non uniform real time traffic

We consider Jain Fairness index [118] to measure how fair our honeynode selection strategies are. For a network of n nodes, if observed values of a performance



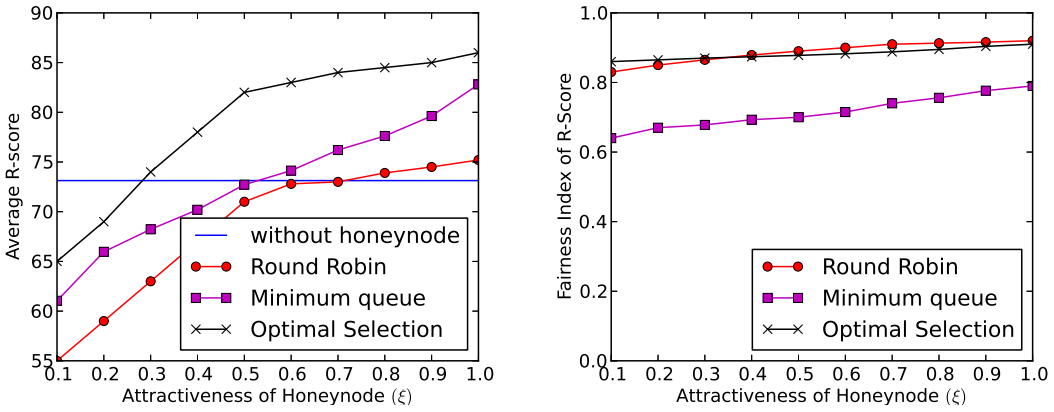
(A) Average queuing delay (B) Fairness Index among all SUs

FIGURE 4.25: Simulation results for CRN with non uniform traffic.

parameter are $x_1, x_2, \ldots x_n$ for n nodes, then fairness index is defined as $\frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n\sum_{i=1}^{n} x_i^2}$. Figures 4.25a and 4.25b depicts the overall average queuing delay for the CRN and fairness index of queuing delay, respectively. All the SUs choose a λ randomly from the range given in the X-axis. For example, in the third simulation, all SUs have λ in between 0.6 and 0.8. The first three simulation sets have lower variances of λ among SUs. The last set of simulations have higher variances of λ. Here we consider FTP data transfer as the application. We can clearly see that the minimum queue honeynode selection strategy provides a lower queuing delay, compared to other honeynode selection strategies. However, the minimum queue honeynode selection strategy performs very poorly, in terms of fairness. Actually, some SUs get better transmission by making the SUs that have lower λ to starve of packet transmission.

4.7.7 Performance of CRN for real time non-uniform traffic

In this section we study how the CRN can achieve fairness for all SU's utility. We have already stated that utility or system performance for real-time traffic is dependent only on PDR as it is not stringent to delay. Eq. 4.15 shows that PDR



(A) Average R-Factor among all SUs (B) Fairness Index of all SUs

FIGURE 4.26: Comparison of performance for all honeynode selection strategies

only depends on ξ. Now, ξ depends on the efficiency of the learning mechanism used in the honeynet. Selection of honeynode does not have any effect on ξ or PDR. So, for real-time traffic selecting honeynode with minimum queue strategy is optimal as it achieves the lowest average queuing delay and thus highest utility.

As a first step towards analyzing CRN with non-uniform traffic, We consider VoIP traffic to study the performance of real-time-traffic. We run sets of simulation where all SUs in the CRN transmit VoIP data in accordance with different voice codecs chosen randomly from the Table 4.7. For simplicity in the simulation we have excluded the codecs with active voice detection and silence suppression. All codecs demand different throughput and have different packet arrival rates.

We have proposed a optimal honeynode selection algorithm in section 4.6.2. For VoIP traffic we consider R-Score as the performance measure or utility function in eq. 4.16. It is interesting to know that all codecs result in different R-score for same PDR and delays. Figure 4.26a shows the average R-score observed for the CRN which reveals that our proposed optimal strategy is performing better compared to other honeynode selection strategies. Here we can see that CRN without honeynode results in R-Score of 73.13. We observe that the lowest effective attractiveness (ξ) are 0.283, 0.53, 0.71 for optimal selection, minimum queue

TABLE 4.7: VoIP packet characteristics [8]

Codec	Voice Payload	Packets Per Second	MAC Bandwidth
G.711 (64 Kbps)	160 bytes	50	87.2 Kbps
G.723.1 (5.3 Kbps)	20 bytes	33.33	20.8 Kbps
G.723.1 (6.3 Kbps)	24 bytes	33.3	21.9 Kbps
G.729 (8 Kbps)	20 bytes	50	31.2 Kbps

selection and round robin selection respectively. Figure 4.26b plots the fairness index for honeynode selection strategies. We can see that minimum queue selection strategy performs very poorly while the optimal selection and round robin are pretty fair. Round robin obtains lower fairness index in this simulation as different codecs provide different R-Score even if the SUs achieve same queuing delay. So, we can conclude that the optimal honeynode selection strategy is performing well for real-time VoIP traffic.

4.8 Summary

In this chapter we propose CR-Honeynet, a CRN sustenance mechanism, which exploits the fact that an intelligent and rational attacker aims for certain transmission characteristics to gain highest impact out of jamming. The stochastic learning model presented shows that the honeynet can confidently learn the attacker's strategy and dynamically evolve with attacker's strategy change. The mechanism efficiently lures the attacker towards attacking the active decoy trap and thus bypassing attacks on legitimate SU communications. The state-of-the-art testbed developed using off-the-shelf software defined radios prove the effectiveness of the mechanism. Currently, the mechanism has a drawback of not placing active decoy while it is passively learning attacker's strategy.

As the second step, we have presented a theoretical model to predict the performance of CRN based on queuing model with fixed vacation. The model deals

with the periodic sensing of cognitive cycle as fixed periodic vacation. We show that CR-honeynet is effective to prevent jamming attack; however assigning honeynode without considering queuing delay associated with it causes performance degradation. Under such circumstances we have shown that dynamic assignment of honeynode is crucial from the system's performance perspective. We propose state dependent honeynode selection strategies at the beginning of every transmission cycle where the honeynode selection can be done by choosing the SU that has highest probability of emptying the queue. We have demonstrated that this strategy performs well when all the SUs in the CRN are having identical traffic load. We have also analyzed the fairness of performance when SUs have nonuniform traffic demand. Simulation results reveal that for real-time traffic our proposed honeynode selection strategy provides optimal system performance while maintaining fairness

In the future, we shall investigate more to improve the learning mechanism, where the honeynet would be able to predict the attacker's strategy change and can place an active decoy to mitigate attack.

Chapter 5

Survivability against induced attacks

In the last chapter, we discussed how a CRN could avoid jamming by luring the adversary to an active decoy while the legitimate SU transmit on another channel. In that scenario, we considered the channels to be homogeneous. In this chapter, we particularly focus on a special vulnerability in cognitive radio networks with heterogeneous spectrum bands, known as induced attack. It is a special case of disruptive attack where an attacker uses its intelligence to force secondary users (or networks of users) to leave bands in the CR network. Cognitive radio networks are envisioned to be learning from their environment by observing, taking feedback, and analyzing their benefits from the band(s) through various courses of action. As such, inducing them maliciously through "intelligent" shadow-disruptive attacks has serious and long-term effects. For instance, successful induced attacks on candidate channel(s) will not only force the secondary networks to leave the band(s), but such successful frequent attacks may also harm their capability to learn about the particular bands. Therefore, even though in reality the spectrum bands may be available or even have high payoffs, secondaries will be reluctant to consider these bands as potential candidate channels, thus limiting their available radio

resources. In the presence of such adversarial scenarios, the problems of survivability, network management, and specifying performance bounds become highly challenging and must be addressed, otherwise the performance of the secondary network will be degraded defeating the purpose of the DSA paradigm.

In this research, we formulate the problem as a game between SUs and the induced attacker, where the game is played in a heterogeneous dynamic spectrum access environment. The induced attacker's aim is to not only disrupt the CRN operation but also to move the CRN toward "inappropriate cognition under malicious stimuli". To investigate the conflict, we first demonstrate a static scenario where the usage of channels with different utilities is formulated as a solution of a zero-sum game. We further enhance our formulation for more realistic scenarios, when the channel utilities are no longer stationary and change abruptly (regime change) due to different channel characteristics, primary user, and disruptions that are not known beforehand. Under such dynamic regime changes, it becomes particularly complex to estimate and decide optimally amidst the presence of an induced attacker. To address this difficulty, we adopt a tradeoff strategy between *exploration* and *exploitation* and compare our mechanism to benchmark situation where utilities are known exactly as soon as a regime change is detected.

The main contributions of this chapter are as follows:

- A game-theoretic framework for making choices over channels to maximize channel utility in the presence of malicious induced attacks This includes a closed form solution for optimal CRN strategies for both SU and attacker. We will often refer to SU as simply user.
- A stochastic model for defining, estimating, and learning channel utilities.
- Demonstration by analysis and simulation that the above obtained CRN strategies combined with estimation and learning can still provide adequate

performance when actual utilities are replaced by their estimated values. In addition, we compare our mechanism to other conceivable benchmark strategies and show improved performance.

The rest of this chapter is organized as follows. Section 5.1 describes the IEEE 802.22 based system model, its challenges and the problem statement as the long term maximization of profit given channel utilities (leading to the game theoretic approach). In Section 5.2, we present the game theoretic approach. In section 5.3, we describe a possible notion of utility. We further develop our study in Section 5.4, where utilities are no longer stationary for various reasons, and also not known beforehand. We present our proposed mechanism combining estimation and learning mechanisms with optimal play. Section 5.5 presents the simulation and numerical results.

5.1 Preliminaries and problem setting

5.1.1 System model

The IEEE 802.22 standard committee has aimed to develop the standard for the cognitive radio access strategy [119, 120]. The standard specifies the physical and MAC layer operation of SUs in TV broadcast bands. A typical CR network is a single hop point-to-multipoint wireless networks, in which a central controller controls the resource allocation. Commonly, a base station (BS) determines which spectrum to use after all the customer premises equipment (CPE) under its supervision send the spectrum sensing report. The standard supports dynamic spectrum access where the SUs apply cognitive capability and use spectrum in an opportunistic manner. Both the BS and CPE perform spectrum sensing periodically to sense the presence of PUs. The spectrum sensing reports are fused together to

obtain the spectrum occupancy and availability map for the entire cell. Although the specifications provide strict protection mechanism to keep the incumbent primary users free of interference, it does not confirm suitable protection mechanism for a CR network from another CR system. When multiple unlicensed operators are operating over a small available band of frequency, there is a chance that they will cause interference among themselves [121].

When a CR node switches on or moves to a new frequency channel, it performs listen before talk to detect the presence of the PU as well as BSs within its communication range. Since it is possible for each node in the network to choose its spectrum band, it is necessary for the given CR node to listen to the preferred channels of the BSs. The different physical propagation characteristics of electromagnetic waves over different spectrum bands is another concern for CRNs. A low-frequency signal (e.g., 700MHz) can travel farther, penetrate walls and other obstacles but its information capacity is lower, and the accuracy in determining the direction of arrival is poorer. However, a higher frequency signal (e.g., 5.0GHz) can only travel a shorter distance, but will be able to carry more information and will exhibit better directionality. Thus, the channels provide different reward or utility upon its usage.

5.1.2 Threat model

The "open" philosophy of the CR paradigm makes such networks susceptible to attacks by smart malicious users that could even render the legitimate CR spectrumless. Due to software reconfigurability, CRs can even be manipulated to disrupt other CRNs or legacy wireless networks with greater impact than traditional hardware radios [122].

In the self-coexistence battle for spectrum opportunities, when the secondary networks are already competing for their survival against other secondaries, the effect

of malicious disruptions can be even more fatal as there is no way to understand whether the disruptions are unintentional or intentional. The motivation for such shadow-disruptive attack behavior can be either monopolism, to capture as much spectrum as possible for themselves without maintaining any spectrum sharing etiquette and make other secondaries starve and eventually to go out of the competition; or adversarial – to disrupt other secondaries' communications and shut them down (particularly applicable in environments filled with adversarial users/networks). To defend against such smart disruptions, it is absolutely critical to understand the uniqueness of the attack models/strategies in exploiting the finest granularity of spectrum agility and the shadow-disruptive nature of the malicious societies.

Induced Attack: In this research, we investigate a particular type of disruptive attack where an attacker uses its intelligence to force users to leave bands in the CR network [123]. This is manifested by the awareness of the CR users that a potential attacker may exists, which could be reinforced by sensing disruptions (e.g. one could detect jamming attempts). As such, and due to the presence of these malicious disruptions, a user is often confronted to make a choice over channels (knowing that an attacker is likely to target the "best" ones). Therefore, to optimize some function, the user will have to migrate from using what is perceived as best channel to other channels. This not only affects performance, but also the ability of the user to effectively learn about channels. For example, few dynamic spectrum access algorithms gather channel access statistics for PUs in an attempt to predict when the channel will be idle [124]. Here the learning radios will be hindered by not being able to access their desired bands. Therefore, this degrades the performance of the CRN not only in terms of instantaneous transmission, but also on the long run.

For the remainder of this chapter, the term "user" means a secondary user of the CRN. Similarly, the term "attacker" means a malicious secondary user of the CRN.

For simplicity, the primary user of a channel is assumed to be "protected" and, hence, not affected by any attacks. The role of the primary user in our context is to simply claim or release the channel to be used by the secondary users (which include user and attacker).

Let's call u_j the utility of channel j when a user successfully transmits on that channel. Therefore, a successful transmission over channel j provides its user with a profit of u_j. This utility u_j is a function of certain properties of the channel; for instance, it could be related to the transmission rate. However, if the channel is attacked during transmission, the utility is given by $v_j < u_j$. Typically, u_j is positive and v_j is 0 or negative; however this is not necessary as long as $v_j < u_j$. For instance, v_j could represent the energy loss (negative) in an unsuccessful transmission as a result of the attack. In such a case, the user makes a profit of v_j (instead of u_j). We assume that u_j and v_j have compatible units when they contribute to the profit. The goal of the user is then to maximize the long term profit by making an appropriate choice of channel when transmitting in each slot. The attacker has the opposite goal. The terms "user" and "attacker" are thus defined in this context. Consequently, an "attack" is regarded as any use of a channel by the attacker, i.e. with the intention to reduce the long term profit as described above.

While the model incorporates one user and one attacker, one could think of all users as one by the virtue that channel requests may go through a centralized controller [125, 126]. Similarly, the aggregate behavior of all attacks may be thought of as coming from a single source. A user acting "optimally" in the face of such aggregate attacks does not need to explicitly distinguish the identity of the attacker. Nevertheless, the scenario of multiple users/attackers remains a valid one because "users" become "attackers" when competing for the same resources. This, however, is not the scope of this chapter.

Our problem setting is focused on the following: We assume that the user is aware of the possible presence of an attacker, who in turn is regarded as a rational player. We consider the simple case where only one channel is chosen for transmission (by the user) and only one channel is chosen for an attack (by the attacker). The user and the attacker do not know which action their opponent will make at each time slot. With that in mind, we formulate the problem as a repeated zero-sum game, where both the user and the attacker make their choices over the available channels. The optimal strategy of our game depends on the knowledge of the channel utilities as described above. Realistically, however, such information is often not directly accessible. Thus we use a stochastic approach to estimate and learn the values needed. Both the game-theoretic approach and the stochastic estimation/learning are described in detail in the following sections.

5.2 The proposed game model

We formally study the case where only one channel is used at a time. We avoid the case when multiple channel are used at once to exclude the possibility that all channels can be attacked, which leads to a rather uninteresting scenario; we assume that no attacker has such a power [127–129]. We have n channels available in the CRN, a user, and an attacker. Channel i has two utilities $u_i > v_i$ as explained above. We define $\Delta_i = u_i - v_i > 0$. In every time step, the user chooses a channel i for transmission, and a channel j is attacked. If $i \neq j$, the user accumulates a profit u_i; otherwise, the channel is blocked and the user accumulates a profit v_i. As described in Section 3, the user seeks to maximize the total profit on the long run (while the attacker seeks the opposite).

It is not hard to see that pure strategies, where the choice over the channels is fixed, do not typically lead to Nash equilibrium. For instance, let i and j be the channels for the user and attacker, respectively. If $i \neq j$, then the attacker may

decrease the profit by following the user and redirecting the attack to channel i. Similarly, if $i = j$, then the user may increase the profit by moving away from that channel. This pattern of following and moving away can continue indefinitely from one channel to another, showing that an optimal strategy must be probabilistic in nature. In this case, the user/attacker generally seeks to maximize/minimize the *expected* profit.

In game theory, such a strategy, where each channel is chosen with some probability, is called a mixed strategy, and is the solution of a linear program for the zero-sum game with the following *payoff matrix*, where the entry in the i^{th} row and j^{th} column represents the profit when the user transmits on channel i, and the attacker chooses channel j.

$$\begin{pmatrix} v_1 & u_1 & \cdots & u_1 \\ u_2 & v_2 & \cdots & u_2 \\ \vdots & \vdots & \ddots & \vdots \\ u_n & u_n & \cdots & v_n \end{pmatrix}$$

We will analyze this game and obtain the optimal strategies for both user and attacker (in a Nash equilibrium sense). We will use $p = (p_1, \ldots, p_n)$ and $q = (q_1, \ldots, q_n)$ to denote the probabilities over the choice of channels for the user and the attacker, respectively. The goal is to compute p and q that represent the optimal mixed strategies: Given the optimal q, any deviation from the optimal p will decrease the profit. Similarly, given the optimal p, any deviation from the optimal q will increase the profit. Therefore, both user and attacker have the incentive to maintain these probabilities.

In principle, a knowledge of u and v means that the optimal strategies for the game will be known to both user and attacker; therefore, it does not hurt if they actually announce them. On the one hand, if the user announces a strategy p,

the best response for the attacker will be to choose a channel j that achieves an expected profit of $\min_j p_j v_j + \sum_{i \neq j} p_i u_i$. The user should choose p to maximize the profit against that best response. Therefore, the user's interest is to maximize $\min_j p_j v_j + \sum_{i \neq j} p_i u_i$. This is equivalent to the linear program LP_1.

$$\begin{aligned} \text{maximize} \quad & z \\ s.t. \quad & \forall j, z \leq p_j v_j + \sum_{i \neq j} p_i u_i \\ & \sum_i p_i = 1 \\ & \forall i, p_i \geq 0 \end{aligned} \tag{5.1}$$

On the other hand, symmetrically, if the attacker has to announce a strategy $q = (q_1, \ldots, q_n)$, the best bet is to choose q to minimize the expected profit under the user's best response, in other words, minimize $\max_i q_i v_i + \sum_{j \neq i} q_j u_i$. By observing that $q_i v_i + \sum_{j \neq i} q_j u_i = u_i - q_i \Delta_i$, we obtain the linear program LP_2 (the dual of LP_1):

$$\begin{aligned} \text{minimize} \quad & y \\ s.t. \quad & \forall i, y \geq u_i - q_i \Delta_i \\ & \sum_j q_j = 1 \\ & \forall j, q_j \geq 0 \end{aligned} \tag{5.2}$$

Let z^* and y^* be the optimal solutions for LP_1 and LP_2, respectively. It is known that $z^* = y^*$ by linear programming duality. Therefore, by solving LP_1, the user (maximizer) can determine a strategy for itself that guarantees an expected profit of at least z^*, no matter what the attacker does. And by solving the dual LP_2,

the attacker (minimizer) can guarantee that the expected profit is at most y^*, no matter what the user does. Since $z^* = y^*$, this determines the optimal way to play for both.

What do we expect from the optimal solutions? Intuitively, some channels should never be used. For instance, given two channels i and j with $v_i = 0$, $u_i = 1$, $v_j = 2$, and $u_j = 3$, one would expect that the user should never choose channel i, as channel j offers a better profit, even when under attack. Consequently, the attacker should also have no interest in attacking channel i. On the other hand, a user should typically favor a reliable channel i with a small $\Delta_i = u_i - v_i$, because the attacker cannot dramatically deteriorate its profit. Algorithm 1 illustrates the optimal solutions for p and q and reflects these observations. In an initial phase (lines 4-13), some channels are eliminated. Among the remaining channels, the user assigns in a second phase (lines 14-17) a probability to a channel i that is inversely proportional to Δ_i. We leave the detail of how the solutions are derived to the Appendix.

Algorithm 5: Optimal solutions for LP_1 and LP_2

```
1  S ← [n]
2  p ← 0
3  q ← 0
4  repeat
5      T ← ∅
6      for j ∈ S do
7          x_j ← 1 − (|S|−1+Σ_{i∈S}(v_i−v_j)Δ_i^{-1}) / (Δ_j Σ_{i∈S} Δ_i^{-1})
8          if x_j < 0 then
9              T ← T ∪ {j}
10     S ← S − T
11 until T = ∅
12 for j ∈ S do
13     q_j ← x_j
14     p_j ← Δ_j^{-1} / Σ_{i∈S} Δ_i^{-1}
```

Given a set of channels $T \subset [n]$, we also define the linear program LP_1^T to be the same as LP_1 after dropping all channels $k \in T$, i.e. dropping p_k and the k^{th} constraint for all $k \in T$. This corresponds to LP_1 when the k^{th} row and the k^{th} column for all $k \in T$ are eliminated from the payoff matrix. We define LP_2^T similarly.

5.2.1 A feasible solution for LP_1

A feasible solution for LP_1 can be obtained by making

$$p_1 v_1 + \sum_{i \neq 1} p_i u_i = p_2 v_2 + \sum_{i \neq 2} p_i u_i = \ldots = p_n v_n + \sum_{i \neq n} p_i u_i$$

which requires solving

$$p_i \Delta_i = p_{i+1} \Delta_{i+1}$$

$$\sum_i p_i = 1$$

and yields the solution:

$$p_i = \frac{\Delta_i^{-1}}{\sum_j \Delta_j^{-1}} \tag{5.3}$$

for a profit $z = [\sum_i u_i \Delta_i^{-1} - 1] / \sum_i \Delta_i^{-1}$. In addition, dropping any number of channels can only yield a feasible solution for LP_1.

Lemma 5.1. *A solution for LP_1^T augmented with $p_k = 0$ for all $k \in T$ is feasible for LP_1.*

Proof: The following linear program is LP_1^T.

$$\begin{aligned} \text{maximize} \quad & z \\ s.t. \quad & \forall j \notin T, z \leq p_j v_j + \sum_{i \notin T \cup \{j\}} p_i u_i \\ & \sum_{i \notin T} p_i = 1 \\ & \forall i \notin T, p_i \geq 0 \end{aligned}$$

For a given $l \in T$, the dropped constraint in LP_1 is $z \leq p_l v_l + \sum_{i \neq l} p_i u_i$. When $p_k = 0$ for all $k \in T$, this constraint becomes $z \leq \sum_{i \notin T} p_i u_i$. But since $u_i > v_i$ ($\Delta_i > 0$), this constraint is dominated by any constraint in LP_1^T that replaces in its sum the term $p_j u_j$ with the term $p_j v_j$ for some j, as shown above. ■

5.2.2 A feasible solution for LP_2

Consider a relaxed version of LP_2 by dropping the positive constraint on q (q has been replaced by x below to emphasize that the two are different solutions and for the purpose of maintaining a clear notation):

$$\begin{aligned} \text{minimize} \quad & f \\ s.t. \quad & \forall i, f \geq u_i - x_i \Delta_i \\ & \sum_j x_j = 1 \end{aligned} \tag{5.4}$$

where $q \in [0, \infty)$ has been replaced by $x \in \mathbb{R}$.

Lemma 5.2. *The optimal solution f^* for the above linear program can be obtained by making*

$$u_1 - x_1\Delta_1 = u_2 - x_2\Delta_2 = \ldots = u_n - x_n\Delta_n$$

Proof: To show this, assume that $\sum_i x_i = 1$ and the above equality holds. A better solution will have to decrease $u_i - x_i\Delta_i$ for every i. Since $\Delta_i > 0$, this means x_i must increase for every i, making it impossible to maintain $\sum_i x_i = 1$. ■

With the addition of $\sum_j x_j = 1$, the equality in Lemma 2 yields:

$$x_j = 1 - \frac{(n-1) + \sum_i (v_i - v_j)\Delta_i^{-1}}{\Delta_j \sum_i \Delta_i^{-1}} \tag{5.5}$$

for $f^* = [(n-1) + \sum_i v_i\Delta_i^{-1}]/\sum_i \Delta_i^{-1}$. Observe that $y^* \geq f^*$ (because x is less constrained than q), but if $x \geq 0$, then $y^* = f^*$ and we have an optimal solution for LP_2.

Lemma 5.3. *Let $T = \{k|x_k < 0\}$. A solution to LP_2^T augmented with $q_k = 0$ for all $k \in T$ is feasible for LP_2.*

Proof: The following linear program is LP_2^T.

$$\begin{aligned} \text{minimize} \quad & y \\ s.t. \quad & \forall i \notin T, y \geq u_i - q_i\Delta_i \\ & \sum_{j \notin T} q_j = 1 \\ & \forall j \notin T, q_j \geq 0 \end{aligned}$$

Let $l = \max_{k \in T} u_k$. Since $x_l < 0$ and $\Delta_l > 0$ and $f \geq u_l - x_l \Delta_l$, it follows that $f^* > u_l$; therefore we have $y^* \geq f^* > u_l$. For a given $k \in T$, the dropped constraint in LP_2 is $y \geq u_k - q_k \Delta_k$. When $q_k = 0$, this constraint becomes $y \geq u_k$, which is dominated by the same constraint when $k = l$. Therefore, to prove the claim, we need to show that the dropped constraint $y \geq u_l$ is dominated by another in LP_2^T, i.e. that $u_l \leq u_i - q_i \Delta_i$ for some $i \notin T$. Assume $u_l > u_i - q_i \Delta_i$ for all $i \notin T$, then we have a feasible solution $y = u_l$ for LP_2, a contradiction since $y^* > u_l$. ■

The analysis above suggests the following approach for finding a feasible solution for LP_2: Given the optimal solution to the relaxed version of LP_2 (4), drop all channels in $T = \{k | x_k < 0\}$, thus obtaining a new instance LP_2^T with a smaller number of channels. Do this repeatedly until no channel j satisfies $x_j < 0$. Finally, solve LP_2 by making $q_j = x_j$ for every $x_j \geq 0$ and $q_j = 0$ for every dropped channel j.

5.2.3 The optimality of both solutions

Drop from LP_1 exactly those channels that are dropped from LP_2, and let S be the set of channels that remain, observe that

$$z = \frac{(\sum_{i \in S} u_i \Delta_i^{-1}) - 1}{\sum_{i \in S} \Delta_i^{-1}} = \frac{(|S| - 1) + \sum_{i \in S} (u_i \Delta_i^{-1} - 1)}{\sum_{i \in S} \Delta_i^{-1}}$$

$$= \frac{(|S| - 1) + \sum_{i \in S} v_i \Delta_i^{-1}}{\sum_{i \in S} \Delta_i^{-1}} = y$$

Thus the feasible solutions for LP_1 and LP_2 become optimal by linear programming duality and, therefore, the above is also equal to z^* and y^*.

Theorem 5.4. *The optimal solutions for LP_1 and LP_2 can be computed by Algorithm 5 below with $u > v$, and n as input, for a profit of $z^* = y^* = [\sum_{i \in S} u_i \Delta_i^{-1} -$*

$1]/\sum_{i\in S}\Delta_i^{-1}$, *where* S *is the set of channels* i *with* $p_i > 0$.

5.3 A notion of utility

Regardless of how channel utility is obtained, it is realistic to assume that such information is not readily available. In this section, we develop a stochastic model for utility. We assume here a standard paradigm of sensing and transmission, though the detail is irrelevant for the theoretical treatment herein. In principle, a user can sense the channel to obtain some properties of the channel, such as availability and transmission rates. Consequently, the channel utility as perceived is related to a particular characteristic of the transmission over that channel. During the t^{th} transmission period, the user observes an instantaneous profit $r_j(t)$ for the chosen channel j, when transmission is successful.

Let $I_j(t)$ be the indicator of the event that the user successfully transmits on channel j during slot t. In other words, if $c(t)$ is the channel chosen by the user at time t, and $a(t)$ the one attacked, then $I_j(t) = \mathbf{1}_{\{c(t)=j, a(t)\neq j\}}$. Similarly, let $A_j(t) = \mathbf{1}_{\{c(t)=a(t)=j\}}$. Define $n_j(t) = \sum_{m=1}^{t} I_j(m)$. In the *stationary* operation of the system, the choice of channel is independent of the time slot, and the utility u_j can be defined as a (stationary) expectation conditioned on a successful transmission:

$$u_j = \mathbb{E}[r_j(m) \,|\, I_j(m) = 1].$$

The statement below is a standard consequence of the law of large numbers, but we present it for completeness and to introduce a method for estimating u_j.

Lemma 5.5. *Under stationary operation of the model, the utility satisfies*

$$u_j = \frac{\mathbb{E}\left[\sum_{m=1}^{t} r_j(m) I_j(m)\right]}{\mathbb{E}[n_j(t)]} = \lim_{t\to\infty} \frac{\displaystyle\sum_{m=1}^{t} r_j(m) I_j(m)}{n_j(t)}, \quad \textit{w.p.1.} \tag{5.6}$$

Proof: Consider the following equality:

$$\begin{aligned} \mathbb{E}[r_j(m) I_j(m)] &= \mathbb{E}[r_j(m) \times 1 \,|\, I_j(m) = 1]\mathbb{P}(I_j(m) = 1) \\ &+ \mathbb{E}[0 \,|\, I_j(m) = 0]\mathbb{P}(I_j(m) = 0) \end{aligned}$$

Therefore, the channel's utility is defined by

$$u_j = \mathbb{E}[r_j(m) \,|\, I_j(m) = 1] = \frac{\mathbb{E}[r_j(m) I_j(m)]}{\mathbb{P}(I_j(m) = 1)}$$

Under stationary operation, and from the definition of $n_j(t)$, it follows that (linearity of expectation)

$$t\mathbb{P}(I_j(m) = 1) = \mathbb{E}[n_j(t)],$$

Therefore,

$$\begin{aligned} u_j &= \frac{\mathbb{E}\left[r_j(m) I_j(m)\right]}{\mathbb{E}[n_j(t)]/t} \\ &= \frac{\mathbb{E}\left[\sum_{m=1}^{t} r_j(m) I_j(m)\right]}{\mathbb{E}[n_j(t)]}, \end{aligned}$$

where $\mathbb{E}[r_j(m) I_j(m)] = \mathbb{E}[\sum_{m=1}^{t} r_j(m) I_j(m)]/t$ is given by the linearity of expectation.

The final step uses the strong law of large numbers for stationary processes to establish the almost sure convergence of both numerator and denominator:

$$\mathbb{E}[r_j(m)I_j(m)] = \lim_{t\to\infty} \frac{\sum_{m=1}^{t} r_j(m)I_j(m)}{t},$$
$$\frac{\mathbb{E}[n_j(t)]}{t} = \mathbb{E}[I_j(m)] = \lim_{t\to\infty} \frac{n_j(t)}{t}.$$

Because both numerator and denominator converge with probability one, the ratio also converges with probability one, which establishes the result. ■

Therefore,

$$u_j = \frac{\mathbb{E}\left[\sum_{m=1}^{t} r_j(m)I_j(m)\right]}{\mathbb{E}[n_j(t)]}$$

is the quantity that we wish to estimate and, furthermore, it is worth noticing that for a finite t,

$$\mathbb{E}\left[\frac{\sum_{m=1}^{t} r_j(m)I_j(m)}{n_j(t)}\right] \neq u_j,$$

The expression $\hat{u}_j = \sum_{m=1}^{t} r_j(m)I_j(m)/n_j(t)$ is the one used to estimate u_j (at infinity they are equal with probability 1), thus we have a biased estimator because $\mathbb{E}[\hat{u}_j] \neq u_j$. If $\mathbb{P}(I_j(m) = 1)$ is known, then an unbiased estimation can be obtained by replacing the (random) denominator $n_j(t)$ by its expectation $t\mathbb{P}(I_j(m) = 1)$. Typically, however, such information is hard to obtain in a dynamic environment, so we stick to the biased estimator (see Section 6).

Under induced attacks, if a chosen channel j is attacked, then a collision happens and the transmission fails, and for simplicity, we now assume that a deterministic loss $e_j \geq 0$ is incurred due to the unsuccessful utilization of the channel. In this case, we set $v_j = -e_j$.

While u_j and v_j (or more precisely their estimates) will determine the decisions of the players (as described in Section 4), the actual measured instantaneous profit of the user at a given time t is given by $r_j(t)I_j(m) - e_j(t)A_j(m)$, and the average

profit up to time t is

$$G = \frac{1}{t} \sum_{m=1}^{t} \sum_{j=1}^{n} [r_j(m) I_j(m) - e_j(m) A_j(m)] \tag{5.7}$$

In the following section, we look at the case where the utilities u_j are estimated dynamically from the instantaneous values of $r_j(t)$ with noise, and we incorporate into the model the possibility that these values may undergo abrupt changes, as is the case for example when a primary user PU claims or liberates a channel.

5.4 Dynamic stochastic model

In the previous section, the game model assumes that the profit of accessing a channel is constant and known to both the user and the attacker. However, in reality, the utilities may vary over time due to noise and other considerations, such as the PU's utilization of the channel. Nevertheless, a player who does not know the actual utility is typically able to estimate it after accessing the channel. In the following section, we describe how the user and the attacker can estimate the channel utilities in a realistic manner without any prior knowledge.

5.4.1 Statistical learning for dynamic estimation of utilities

If the expected utilities u_j and v_j were known exactly, then both user and attacker would be best using the optimal strategies found in Section 4. We assume here without loss of generality that v_j is constant for all channels ($v_j = 0$ is our simulation) and we now use the notation $p(u)$ and $q(u)$ for the optimal strategies to make it explicit that they depend on the values of the expected utilities u.

At each time slot, the secondary users first sense the number of available channels n (although this number depends on t, we use n instead of $n(t)$ for ease of notation). Consider a stationary random strategy of the form

$$\theta_j = \mathbb{P}(c(t) = j); j = 1, \dots n$$
$$\alpha_j = \mathbb{P}(a(t) = j); j = 1, \dots, n$$

where $c(t)$ and $a(t)$ are as defined in Section 5. During time slot m, when $I_j(m) = 1$ (also defined in Section 5), the user will measure the instantaneous profit

$$r_j(m) = u_j + \eta_j(m) \tag{5.8}$$

where $\{\eta_j(m) : m \geq 1\}$ are iid zero-mean random variables with bounded variance, representing the noise in actual observations of the channels' properties. Given a stationary regime, the user has an estimate of u_j at time t:

$$\hat{U}_j(t) = \frac{1}{\theta_j(1-\alpha_j)t} \sum_{m=1}^{t} r_j(m) I_j(m), \tag{5.9}$$

This would be an ideal unbiased estimator because $E[\hat{U}_j] = u_j$ by Lemma 1 and the fact that $E[n_j(t)] = \theta_j(1-\alpha_j)t$. However, if the user does not know α_j (because the attacker's estimate of the utilities is unknown to the user), then the user will estimate u_j at time t as the sample average (as described in Section 5):

$$\hat{u}_j(t) = \frac{1}{n_j(t)} \sum_{m=1}^{t} r_j(m) I_j(m), \tag{5.10}$$

Although this latter estimator (5.10) is biased, it converges with probability one to u_j under a stationary regime as $t \to \infty$ (that is, it is a *consistent* estimator). The reason for the bias lies in that $\mathbb{E}[1/n_j(t)] \neq 1/\mathbb{E}[n_j(t)]$.

Similarly, the attacker defines its own analogous (unbiased and biased) estimators $\tilde{U}_j(t)$ and $\tilde{u}_j(t)$ by exchanging the roles of α_j and θ_j, and $c(t)$ and $a(t)$.

Under a stationary regime, although different, both $\hat{u}_j(t)$ and $\tilde{u}_j(t)$ are consistent estimators of u_j for every j that satisfies $\theta_j \neq 0$ and $\alpha_j \neq 0$ respectively (because user and attacker actually use the channel and, hence, obtain some estimates).

In the real system, however, utilities are not stationary and may occasionally change abruptly due to changes in channel characteristics coming from the PUs. We will assume that detection of channel availability/unavailability is instantaneous: as soon as the user and attacker sense the PU, they reset the current list of available channels. Similarly, when a PU liberates a channel the user and attacker have instantaneous knowledge. However users have no means to know when the utilities themselves change values abruptly (e.g. when PU liberates a channel), except by estimation. This calls for a model for regime changes [24], where monitoring the changes becomes necessary in order to produce accurate statistics. In [130, 131], for instance, we declare a regime change on channel j whenever w consecutive observations fall outside the region $\hat{u}_j(t) \pm 3\sqrt{\widehat{\mathbb{V}\mathrm{ar}}(u_j)}$, where $\widehat{\mathbb{V}\mathrm{ar}}(u_j)$ is the estimated variance of u_j.

Once a regime change is declared, estimation of channel utilities and their variances is reset for all channels. Then choosing the channel more often will result in a faster and more accurate estimation of the channel's utility, and thus also of $p(u)$ and $q(u)$. However, the real goal is to play the game optimally, and not to estimate optimally. That is, we wish to estimate $p(u)$ and $q(u)$ accurately so that the actions are chosen precisely by setting $\theta = p(u)$ and $\alpha = q(u)$. Unfortunately, using estimates in real time does not ensure convergence, since $p(\hat{u}_j)$ or $q(\tilde{u}_j)$

might be zero for some j. Specifically, using

$$\theta_j(t) = p(\hat{u}_j(t))$$
$$\alpha_j(t) = q(\tilde{u}_j(t))$$

may lead to a very bad strategy, and it will not converge to the real values if at the moment of a regime change the channel in question satisfies $p(\hat{u}_j) = 0$. In this case, channel j will not be chosen for transmission and, consequently, the estimation will never be updated. We adopt a trade-off strategy between exploration and exploitation, as described below:

As soon as a regime change is detected for any channel, the user/attacker declares a *learning period* of length T time slots (exploration). At this point, the algorithm resets $t = 0$ and uses uniform probabilities for sampling the channels. After the training period is over, we revert to the probabilities given by the newly available estimates $p(\hat{u})$ and $q(\tilde{u})$ (exploitation).

For simplicity, our simulations limit regime changes to changes in PU activity. The following section provides simulation results to evaluate our game-theoretic solution and stochastic estimation mechanism against other reasonable benchmark strategies. Algorithm 6 provides the simulation pseudocode for the user. The attacker also goes through similar pseudocode except it uses its own channel estimation and probabilities, i.e. T_c is replaced by T_a, $\hat{u}_i$ by $\tilde{u}_i$, and p_i by q_i.

5.5 Simulation results and discussions

For the simulator, a sensing period followed by a transmission period form a *slot*. The user and the attacker are synchronized for the sensing and transmission periods.

Algorithm 6: Learning algorithm for user

```
learning ← T_c
while True do
    if change in PU activity then
        learning ← T_c
        t ← 0
    t ← t + 1
    if learning > 0 then
        access available channels with equal probability
        update û_i as in (5.10)
        learning ← learning − 1
    else
        access channel i with probability p_i from Alg. 1
        update û_i as in (5.10)
```

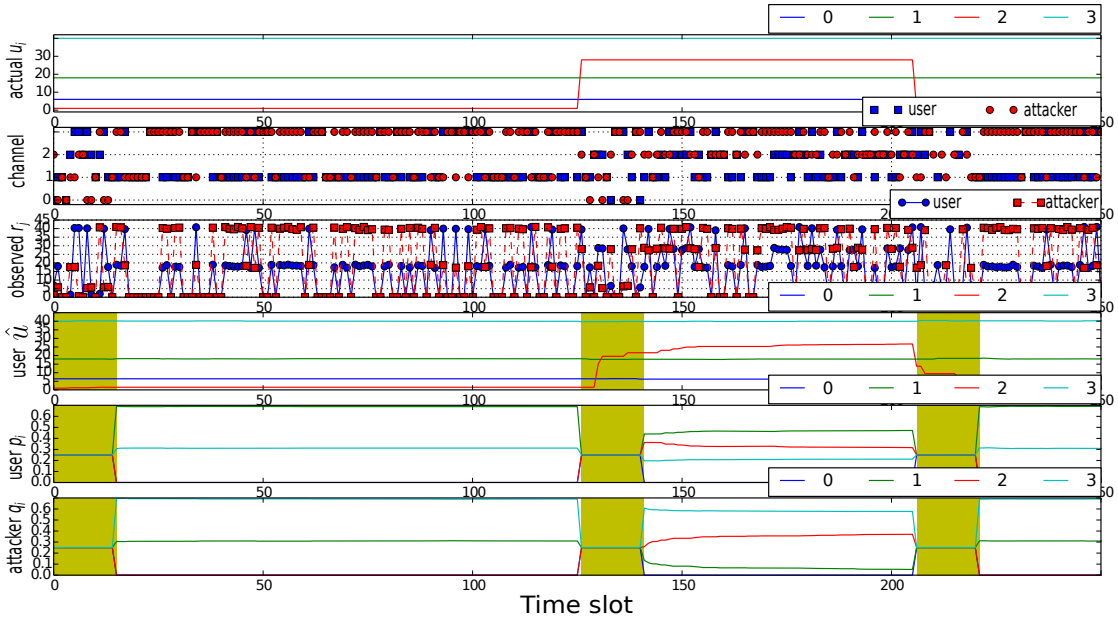


FIGURE 5.1: Depiction of different parameter for a pilot simulation

The total number of channels is fixed. For each channel, the activity of its PU is modeled as an on-off process (each experiment will describe the parameters for that process).

The utility for a channel (u_j) is 0 when its PU is present, as channel j is not accessible within the CRN by any secondary user. Both the user and the attacker can sense the spectrum during the sensing period of each slot and generate a list of available channels to access. They choose one channel to access at the starting

of the transmission period. That choice remains persistent during the entire time slot.

If the channel j is accessed at time slot m without any collision, then a profit $r_j(m)$ as described in (5.8) is observed. This is true for both user and attacker. For simplicity, the simulator uses η_j as a uniform random variable $\sim U(-1, 1)$ for all channels and utility values. That is, the sequence $\eta_j(m)$ consists of independent and identically distributed zero-mean uniform random variables. Both the user and the attacker observe $v_j = 0$ when they collide. They both keep track of the channels used and their observed profits in order to estimate $\hat{u}_j(m)$ and $\tilde{u}_j(m)$ in accordance with (5.10).

Figure 5.1 provides a snapshot of a typical simulation, where the x-axis represents time slots. The PU process was as follows: The number of time slots that a PU stays active is a uniform random variable on $[50, 150]$. The number of time slots without PU activity is independent of previous activity and is uniform on $[50, 600]$. The colors blue, green, red, and cyan correspond to channels 0, 1, 2 and 3 respectively.

The first subplot sketches the actual utility $u_j(m)$ of channel j during slot m. In this subplot, we can see abrupt changes in the utility of channels due to PU arrival or departure. When a channel is being used by the PU, it's utility is 0 as described earlier.

The second subplot shows the channels accessed by the user and the attacker in each time slot. Blue circles indicate the channels used by the user, and red squares trace the channels used by the attacker.

The third subplot reports the observed profit $r_j(m)$ for the user. If at time slot m, the user and the attacker access the same channel j, the user observes $v_j(m) = 0$; otherwise, $r_j(m) = u_j(m) + \eta_j(m)$; where $\eta_j(m)$ is the noise.

The fourth subplot illustrates the estimated utility $\hat{u}_j(m)$ of the user for the different channels during the simulation. A yellow shaded region indicates a learning period. It can be observed that the estimated utilities change abruptly during learning periods, and slowly otherwise.

The fifth subplot portrays the probabilities $\theta_j(m)$ with which the user accesses channels at the given time slot. During a learning period, $\theta_j(m)$ is uniform for all channels; otherwise, the user chooses channels using $\theta_j(m) = p(\hat{u}_j(m))$ during the exploitation phase. The effect of induced attack in this subplot can be clearly observed. Although channel 3 has the highest utility, the attacker forces the user to choose channel 3 with lower probability as dictated by the strategy.

The sixth subplot depicts the channel probabilities $\alpha_j(m)$ for the attacker, in a way similar to the fifth subplot.

In this simulation, we have used a synchronized attacker, i.e. the learning period for both the user and the attacker start and end simultaneously (here $T_c = T_a = 30$). Our basic assumption is that both the user and the attacker can sense PU arrival and departure at the same time. This assumption makes them start and end the learning period in a synchronized way.

We outline below several experiments based on the general simulation framework thus described.

5.5.1 Experiment 1: Learning times

Under the presence of noise, both the user and the attacker enter a learning period to estimate channel utilities as soon as they detect changes in PU activity. In this experiment, we investigate how the length of the learning period affects the overall performance. In particular, we explore the possibility of an equilibrium point (T_c, T_a), from which both user and attacker do not wish to deviate.

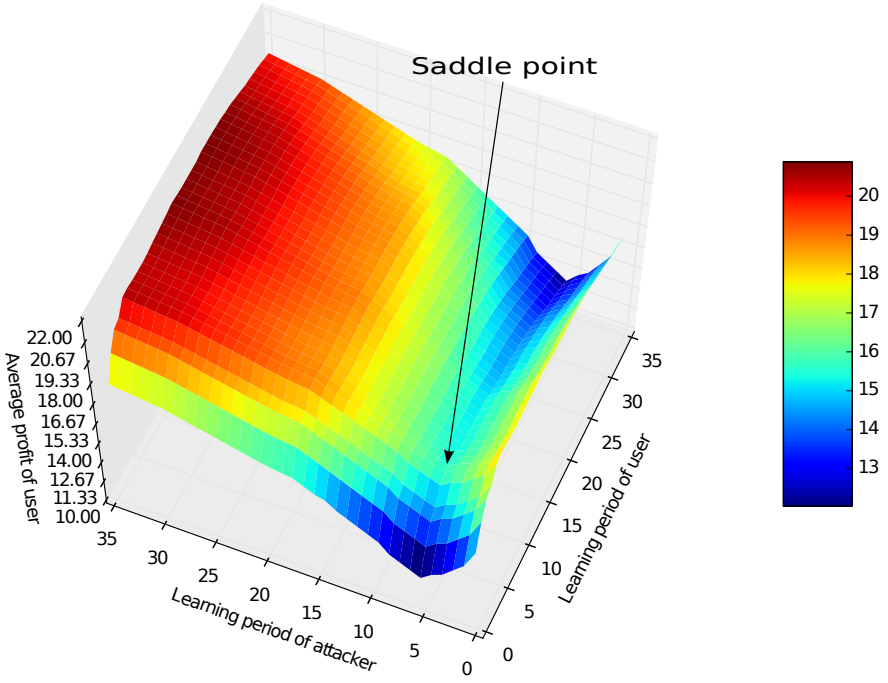


FIGURE 5.2: Saddle point for learning period

The simulator uses 4 channels with utilities 6, 18, 28 and 40. The number of time slots that a PU stays active is a uniform random variable on $[25, 100]$. The number of time slots without PU activity is independent of previous activity and is uniform on $[100, 400]$.

For each combination of user-attacker parameters (T_c, T_a), we run 25 simulations of 100,000 time slots each to obtain averages. Figure 5.2 shows the average profit of the user with different learning periods in a 3-dimensional plot as a function of T_c and T_a. The average profit is calculated as in (5.7) for $t = 100,000$.

As before, the probability of a channel is uniform in a learning period, and is obtained by $p(\hat{u})$ and $q(\tilde{u})$ for the user and attacker respectively during exploitation. The probabilities computed by one player are unknown to the opponent.

On the one hand, if the user has long learning periods, it accesses all channels uniformly for long times, which does not allow the user to effectively explore the better channels. On the other hand, short learning periods do not enable the user to learn the channel utilities properly.

The same observation holds in case of the attacker. Longer learning periods do not allow the attacker to effectively jam the better channels, while shorter learning periods do not enable the proper learning of the channel utilities.

In deed, we observe that there is a saddle point (T_c, T_a), as indicated in Figure 5.2; any deviation from this point hurts the player in question; a deviation in T_c decreases the average profit, and a deviation in T_a increases it. Figure 5.2 reveals the optimal learning period of 6 time slots for both the user and the attacker given the particular simulation parameters. In general, they need not be the same. An approach for computing (T_c, T_a) is beyond the scope of our research. Practically, the learning time may be adjusted in an ad-hoc way until a better performance is observed. The following section illustrates the change in performance as a function of T_c.

5.5.2 Experiment 2: A stronger adversarial model

Following up on the previous experiment, we consider a stronger adversarial model in which the attacker has accurate information of the utilities u_j, and thus does not require a learning mechanism to estimate them. We call this here the *game with accurate information*, in contrast to *estimated information* as before. For this simulation, we fix $T_a = 6$ as obtained above and vary T_c. The plot in Figure 5.3 of the *game with estimated information* is consistent with the previous results, showing a peak for the average profit when $T_c = 6$. The *game with accurate information* is not a practicality, even when the accurate information is on the attacker side, but it represents a worst-case scenario for the user and provides a conservative view of performance.

Naturally, the length of the optimal learning period varies with the number of channels, the frequency of PU arrival/departure, and the mean and variance of channel utilities (given by u_j and η).

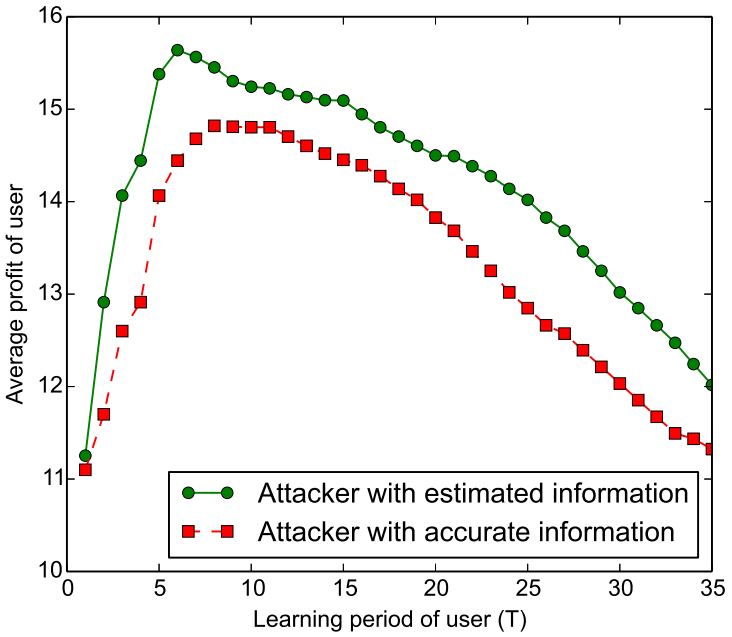


FIGURE 5.3: Performance when attacker has estimated/accurate information. The learning time for attacker is fixed.

5.5.3 Experiment 3: benchmark strategies

To evaluate the efficiency of the game model, we compare it with other typical channel selection schemes for a number of channels $n \in [2, 20]$. The channel utilities are generated for each channel independently and uniformly at random in $[6, 30]$. The number of time slots that a PU stays active is a uniform random variable on $[75, 125]$. The number of time slots without PU activity is independent of previous activity and is uniform on $[300, 500]$.

In the *Random* channel selection, both the user and the attacker choose channels uniformly at random (with equal probability). In the *Greedy* channel selection, the probability of selecting a channel is higher for the channels with higher utility. In this scheme, if n channels are available, they are ranked by increasing utility from 1 to n. The probability of selecting channel j is $j/\sum_{i=1}^{n} i$.

The Random selection scheme does not require knowledge of u_j; therefore, to present a fair comparison among the different schemes, we may assume that both

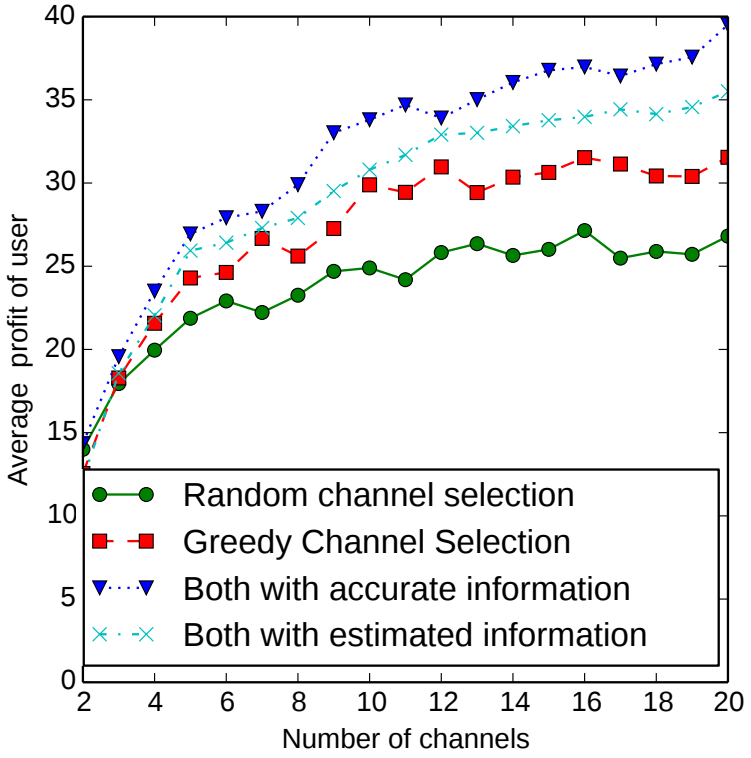


FIGURE 5.4: Comparison of average profit to random and greedy benchmarks

the user and the attacker have accurate information of channel utilities, i.e. no learning periods are required. We simulate the Random and the Greedy schemes with accurate information, and our game model with both accurate and estimated information.

Figure 4 shows the simulation results for the four schemes. With the Random scheme, the user does not utilize the best channels frequently enough; even a random attacker is sufficient to expose this deficiency. Greedy performs better, but the greedy attacker can still frequently prevent the user from successfully utilizing the best channels. Our game model with accurate information based on computing p and q performs the best.

Finally, our game model with estimated information is indeed the scenario of interest, where both user and attacker use learning to estimate utilities. The optimal learning periods T_c and T_a for the different number of channels were computed

by separate simulations. Figure 4 shows that our game model with estimated information falls between its accurate information counterpart and Greedy.

5.6 Summary

In this chapter, we used a game to model the actions of choosing channels for transmission (by the user) and for attack (by the attacker). We described a closed form solution for the game when the channel utilities are known and fixed. The generalization to non-stationary channel utilities is necessary in order to realistically model the PU activity and the uncertainty in obtaining the values of those utilities. In our formulation, we assume that channel utilities can change abruptly, not only by the activity of the PU, but also due to other factors such as channel deterioration, and we allow for the monitoring of regime changes (although for simplicity we only allow changes due to PU activity in our simulation).

Because the optimal game strategies depend on utilities, uncertainty in the estimation of the utilities may introduce significant bias in the operation of the game when the (noisy) estimates are used directly to compute the probabilities p and q for the user and the attacker, respectively. In particular, just after a regime change, channels may have bad estimates of their utilities. In this case, using the closed form solution as if the estimates were the true values may lead to very bad strategies. Instead, we explore a mechanism to learn the new values of the utilities before using the closed form solutions. This mechanism relies on accessing channels in a uniform way during a learning period T. Our simulation results show that the game theoretic solution (exploitation phase) combined with the estimates of utilities through learning (exploration phase), lead to improved results when compared to some benchmark strategies.

Chapter 6

Defense against jammers moving in 3D

So far we have considered networks that are deployed over a small area with limited mobility. In this chapter, we focus on networks with mobility in 3D. We investigate the applicability of adaptive beam forming antennas for spatial filtering which creates a null gain towards a jammer. With the proposed beamnulling approach nodes inside jammed region can also communicate with neighbors without requiring additional resources.

An antenna directs the energy with different gain in different directions in terms of θ (azimuth) and ϕ (altitude). Figure 6.1 illustrates the logical circuitry of a beamforming antenna array. Each element processes the desired signal mixed with interference and noise. Different weights are assigned on each element by the control process in order to create the desired gain pattern. In case of ANA, the weights are assigned in such a way that the radiation pattern creates a null in the desired direction. Once the desired angular direction and the width of the null are determined, the beamformer calculates the weight values for creating the desired antenna pattern with null.

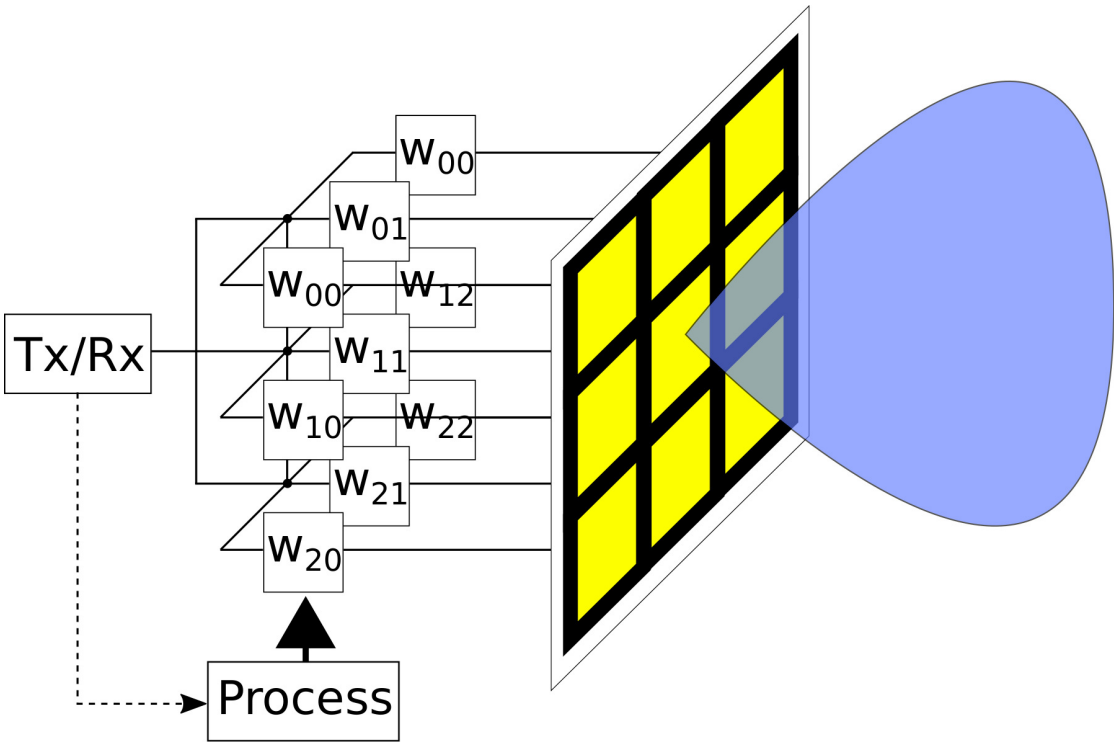


FIGURE 6.1: Schema for ANA

The rest of the chapter is organized as follows: In Section 6.1 we describe the proposed system model. We approach the problem of distributed adaptive beamnulling in two phases. In the first phase, we consider that the beamforming antenna can determine the beamnull borders in elevation and azimuthal angles; thus creating rectangular null region. The methodology and its evaluation is presented in Section 6.2. In the second phase, we optimize the created beamnull using filtered tracking mechanism. The schema for optimized beamnull is presented in Section 6.3. Both the schemes have been evaluated using a customized simulator as well as network simulator 3. FInally 6.4 concludes the chapter.

6.1 System model

The defending network considered in this study is a multi-hop UAV mesh network and arbitrarily distributed in their operating space. Each node is equipped with an antenna array capable of DoA estimation and beamforming.

The jamming attack is sought to be carried out by one or more entities that continuously transmit high powered signals to cause interference on the same spectrum as the network. If the Signal to Interference and Noise Ratio (SINR) of inter-node communications falls below a threshold, the receiver node is considered as jammed. The threshold value of SINR depends on the MAC protocol as well as the modulations and coding scheme. Since the approach proposed in this network is developed to operate on the physical layer, it remains independent of upper layer protocols such as MAC and routing.

6.1.1 System assumptions

i) The jammer is assumed to be a moving node that transmits a disrupting signal on the same frequency as the ad hoc network.

ii) Each node monitors the DoA of jammer relative to its local coordinates. The current research work considers that a jamming signal can be distinguished from other legitimate transmissions by applying mechanisms proposed such as [73, 132, 133]. Nodes can not determine the distance of a jammer accurately as it would require precise distance measuring hardware.

iii) A node can not determine the DoA of jammer's signal while it is communicating with its neighbors. To determine the jammer's DoA, it goes through a sensing phase at every τ seconds interval. We assume that a node communicates with its neighbors in between sensing intervals. During normal communication phase, nodes can not determine whether an interference is caused by another legitimate node in the network or by a jammer. Estimation of DoA has been widely studied in literature. Proposed algorithms can be broadly classified into beamscan algorithm and subspace algorithm [134–136]. In beamscan methods, a region is scanned with conventional beam and the square of the received signal magnitude is recorded. Minimum Variance

Distortionless Response (MVDR) and root MVDR are two examples of this class [137]. On the other hand, the orthogonality between the signal and noise subspaces are exploited in subspace algorithms. MUSIC, Root-MUSIC and ESPIRIT are among the most efficient subspace DoA estimation algorithms in antenna arrays. A thorough review and comparison of widely used DoA estimation methods has been provided in [138]. The current work does not deal with measuring DoA with actual antenna arrays. Instead it simply assumes that DoA can be measured with an error that follows joint normal distribution over θ and ϕ.

iv) Each node is equipped with a beamforming antenna array, capable of introducing nulls in its originally isotropic radiation pattern. The antenna is considered to be an ideal beam null antenna that poses a gain of 0 in the null region or null cone. Time required to change the beam of ANA is negligible compared to the change in the jammer's position. Beamformers are assumed to have sufficient spatial resolution to form the calculated nulled regions with sufficient accuracy [139–141]. Determination of weights on the antenna elements to create a desired beamform is widely studied in literature. Some of the major weight calculation methods are Dolph-Chebyshev weighting, Least Mean Squares (LMS) and Conjugate Gradient Method (CGM) [142]. In the case of mobile ad hoc networks, in which the directions of desired and interference signals are unknown and vary, Stochastic Search algorithms are applied [143]. Examples of such methods are Gradient Search Based Adaptive algorithms [144–146], Genetic Algorithms [147–149] and Simulated Annealing [150, 151]. Thorough reviews and comparison of beamforming methods and algorithms are provided in [143] and [152]. In our study, we consider that null is a cone in the isotropic beam pattern. The gain inside the null is negligible.

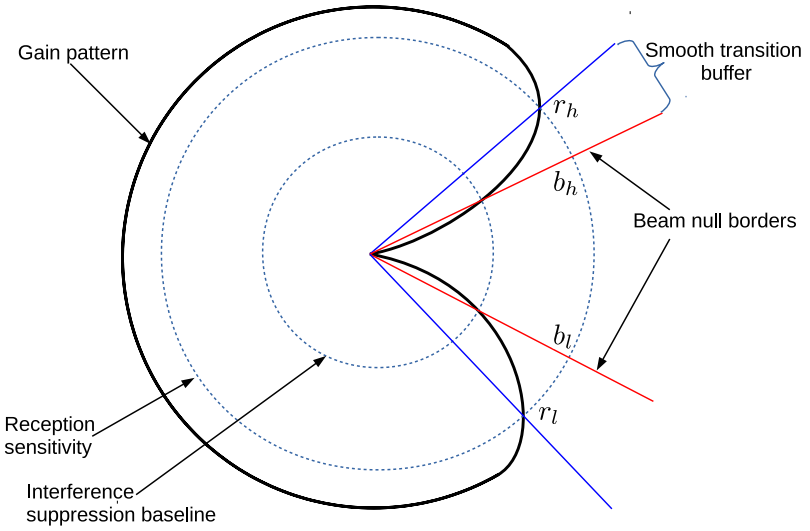


FIGURE 6.2: Depiction of null boundary

v) A link between two nodes fails if either of the two nodes are attacked or one of the nodes fall in the beam null of the neighbor. The MAC layer of the node is unaware of the change in beamform and uses the same protocol as in isotropic antenna. The shadowed neighbors are simply assumed to not be in the range. In this way the complexity and overhead of directional MAC protocol is reduced [153].

vi) Introducing a null in the omnidirectional pattern of a beamforming node may be interpreted as changing the mode of communications to directional transmission, hence necessitating the use of Directional MAC protocols [153]. However, the higher network layers can operate under the default assumption of omnidirectional transmission, as the nulled region is already under jamming and no hidden/exposed terminal problem may arise from its direction [154]. This approach therefore eliminates the overheads associated with most directional communications schemes [155–158].

vii) A beam null is a region in the direction in which the antenna gain is below the cutoff threshold of interference, i.e. the signal arriving in the nulled direction will not cause interference on a node. Figure 6.2 illustrates an example of a gain pattern in 2D and its corresponding null borders. Here, b_h and b_l are

the null borders. Within the receding lobes bounding the nulled region, the gain of received signals falls below the sensitivity threshold, while interference remains above the required cut-off. Hence, the entire transition region is blind to communications, which is accounted for by addition of smooth transition buffers to the beam nulled angle. These regions are defined by borders r_h and r_l. As the gain pattern illustrated in this figure demonstrates, the nulled region is essentially bounded by receding lobes rather than sharp cutoffs. Signal arriving outside of these regions will have full reception. Communication is not possible with neighbors who lie in the buffer or the null region and hence considered as shadowed in the beam null. In the rest of this chapter, we consider the beam null borders to be the boundary in which gain is below interference cutoff i.e. b_h and b_l.

6.2 Rectangular beamnull

6.2.1 Methodology

The proposed framework uses rectangular adaptive beam nulling in order to avoid jamming. Figure 6.3 provides a block representation of the relevant network layers in a node implementing this framework. The jamming detection module uses measured parameters from the medium access control (MAC) and physical layers such as carrier sensing time, packet delivery ratio, signal strength, etc. Various methods for detection of jamming signals have been proposed in the literature, but as the focus of this work is on mitigation of jamming, it is assumed that jamming signals are detectable. Interested readers may refer to [73, 132, 133, 159] for more details on detection techniques. The adaptive beam nulling block uses the DoA measurement of jamming signal and dynamically modifies the beamforming weights of the radio interface to create a null towards the jammer. The upper

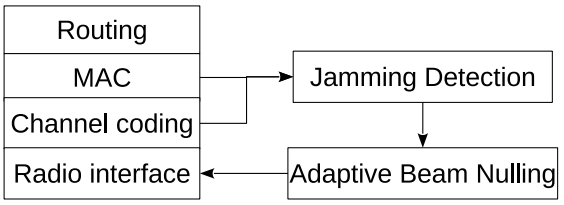


FIGURE 6.3: Block representation of proposed mechanism

layer protocols are unaffected by the beam nulling procedure. If a link fails due to a node falling inside the beam null of its neighbor, the routing protocol treats this as link failure and utilizes an alternative route.

Each node switches to a sensing phase at every time interval of length τ to measure the DoA of jammer's signal. Since the sensing is not continuous, the history of this periodic measurement is then used in the beam nulling stage to predict the movement of the jammer in the time between the current and next sensing phases. Figure 6.4 illustrates an example of DoA measurement in 3D space. In every sensing phase m, the jammer's DoA is measured in terms of its azimuth and elevation angles (θ^m, ϕ^m) in the local coordinate system of the observing node. Let x^m be the observed position of the jammer in the mth sensing phase. The azimuth angle θ^m is then defined as the angle between the X-axis and the projection of the line connecting x^m to the origin on XY plane, and the elevation angle ϕ^m is the angle between the origin–x^m line and its projection on XY.

Using the history of DoA measurements, the jammer's trajectory between the m^{th} and $m+1^{th}$ sensing phases can be efficiently predicted. Consequently, the nulled region is calculated such that it includes the current location of the jammer, as well as its predicted trajectory. Also, since the DoA measurements and predications are both prone to errors, the beam nulling process expands the analytically calculated nulled region by adding a safety zone with the aim of mitigating the effects of errors on nulling the jamming signal. The nulled region can be represented by two boundaries on each of θ and ϕ axes. As is shown in Figure 6.5, the nulled region

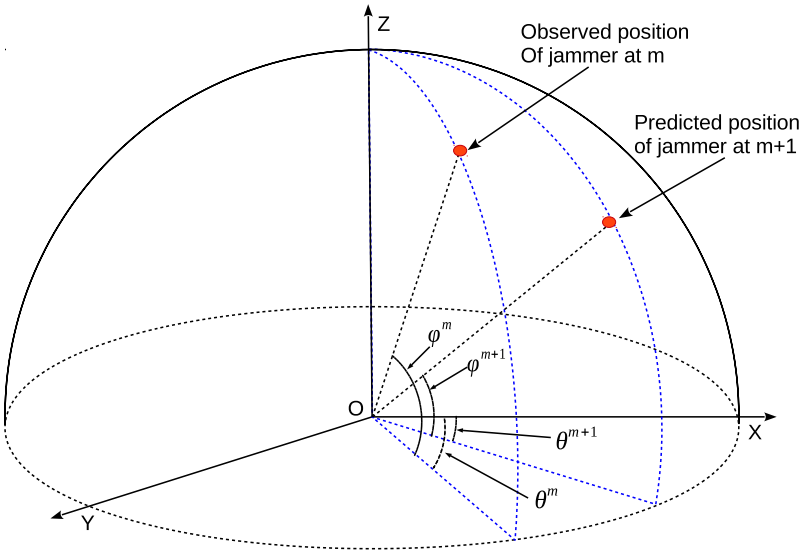


FIGURE 6.4: Observation of DoA of jammer in 3D space

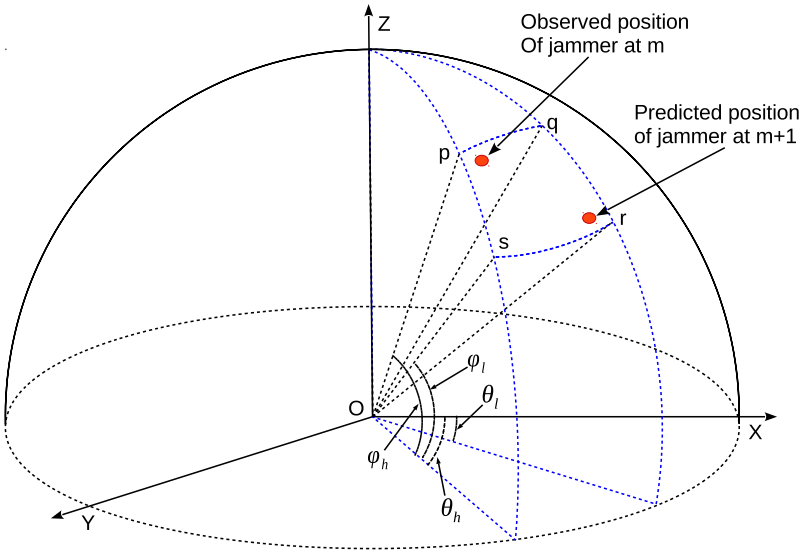


FIGURE 6.5: Depiction of 3D beam null

between the node O and the null cross section $(pqrs)$ can be defined by its borders represented by their corresponding angles θ_l, θ_h and ϕ_l, ϕ_h.

Transmissions from neighboring nodes that fall within the nulled region of a node are also suppressed. Hence, the width of the nulled region must be determined in such a way that it maximizes the confidence in jamming avoidance while minimizing the number of link failures.

6.2.1.1 Problem statement

Let us first look at the problem in a 2-dimensional environment. Figure 6.6 illustrates the effect of adaptive beam nulling in the presence of a moving jammer. In this scenario, the one hop links between node A and its neighbors B, C, D and E are considered. Node A periodically scans for the DoA of the jammer's signal (θ^m) in intervals of (τ) seconds. Due to the discontinuous observation of the jammer's DoA, while calculating the null angle, A must take into account the movement of the jammer between two consecutive observations. This calculation must include prediction of the jammer's angular velocity by considering its history of movements. As the mobility pattern of a jammer becomes more random, the prediction accuracy of its movements decreases. Therefore, the effect of various mobility models of the jammer on a network of beam nulling nodes can provide a practical measure for efficiency of this scheme.

Node A uses a modified beam pattern to communicate with its neighbors until the next sensing period. In Figure 6.6a, A has a narrower null angle compared to Figure 6.6b. With this narrow null angle, A can communicate with B, D and E, whereas with a wider null angle, A can communicate only with B and D. By the next sensing period $m + 1$, the jammer moves to a new position, falling outside of the narrower null, which consequently exposes A to the jammer. As a result, all of A's links are disrupted. On the other hand, the wider null angle maintains the jammer inside the nulled region for the whole interval. The trade-off for widening the null to cover the jammer's probable movements, is the cost of disabling unaffected links. Hence, another important factor in efficiency of beam nulling is the choice of optimum nulling angle in dynamic scenarios.

The practical limitations of adaptive beam nulling, such as inaccuracy in estimation of DoA, as well as hardware limitations in implementing a desired antenna

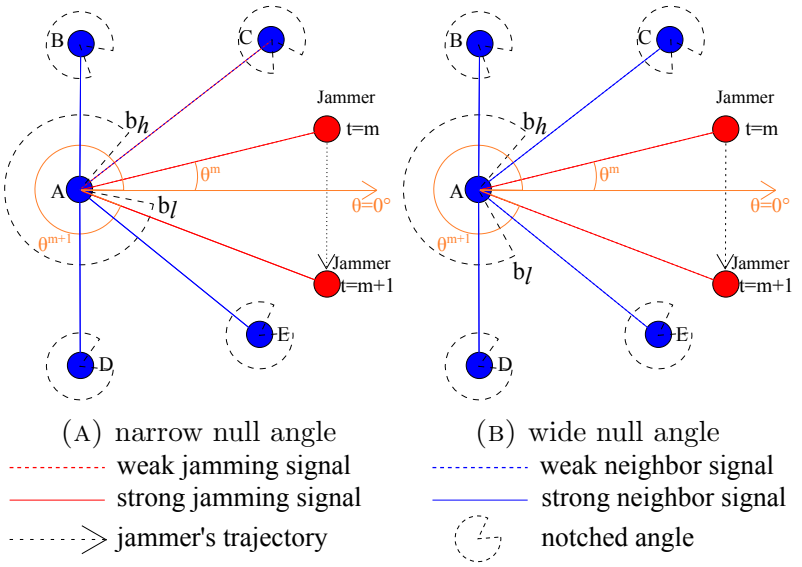


FIGURE 6.6: Depiction of the beam nulling principle.

pattern, lead to introduction of errors in a beamformer's performance. The *measurement error* is the error in DoA estimation. If $(\widehat{\theta_a^m}, \widehat{\phi_a^m})$ is the actual angular position of the jammer with respect to node A, but the observed DoA by A is (θ_a^m, ϕ_a^m), we can write

$$\begin{bmatrix} \theta_a^m \\ \phi_a^m \end{bmatrix} = \begin{bmatrix} \widehat{\theta_a^m} \\ \widehat{\phi_a^m} \end{bmatrix} + \mathbf{e}_{doa} \tag{6.1}$$

where $\mathbf{e}_{doa}$ is the measurement noise with known covariance. Similarly, error is incurred while implementing the beam null border is called beamforming error. Let us say a node calculates a beam null border at $\widehat{b^m}$ and the actual implemented border is at b^m, then we can write

$$b^m = \widehat{b^m} + e_{bn} \tag{6.2}$$

For a sensible study on the efficiency of practical implementation, investigating the impact of system errors in the simulation is of crucial importance. In the subsequent sections we present a framework that determines the beam bull borders

dynamically by incorporating the randomness in the mobility of the jammer as well as hardware limitations.

6.2.1.2 Calculation of null borders in 2D

This section presents a framework for determining the beam null borders in 2D environment. Each node in a multihop ad hoc network uses this method to create a beam null in a distributed manner according to its own frame of reference. After sensing the presence of a jammer, a node i observes the angular position of the jammer or the angle of attack (θ_a^m) with its frame of reference at every sensing phase $m \in \{1, ..., M\}$. Node i then adjusts its beamform to attenuate the jamming signal and communicate with its neighbors until the next sensing phase $(m+1)$. In Figure 6.6, at the m^{th} sensing phase, the jammer is sensed at angle θ_a^m. In the next sensing phase $(m+1)$, i senses the jammer at θ_a^{m+1}. Since the jammer is moving, it may cross the null of the beamform and node i would be affected by the jamming signal. The aim of adaptive beam nulling is to make sure the jammer stays within the nulled region for the entire time between two consecutive sensing phases. Node i calculates the angular velocity of the jammer (v_a^m) as:

$$v_a^m = \frac{\theta_a^m - \theta_a^{m-1}}{\tau}$$

Consider $\overline{v_a}$ and $\sigma(v_a)$ as the mean and standard deviation of the velocity (v_a) of the jammer, respectively. Node i constructs a beam null using an algorithm that considers the history of jammer's movement. A beam null is defined by two borders: b_l^m and b_h^m which are lower and higher angles respectively. Clearly, $\theta_a^m + \tau\overline{v_a}$ gives the estimated location of the jammer at the $(m+1)^{th}$ slot. Since the actual velocity and direction of the jammer are unknown, the null should be wider in case of sudden change in direction or velocity of the jammer. Change of velocity of the jammer can be estimated with $\sigma(v_a)$. If a jammer changes its

direction or velocity, $\sigma(v_a)$ would be high compared to the case when the jammer moves at the same direction with constant velocity. An estimation for the beam null angle can be calculated as:

$$b_h^m = \max(\theta_a^m,\ \theta_a^m + \tau(\overline{v_a} + \alpha\sigma(v_a))) \tag{6.3}$$

$$b_l^m = \min(\theta_a^m,\ \theta_a^m + \tau(\overline{v_a} - \alpha\sigma(v_a))) \tag{6.4}$$

$$\psi^m = b_l^m - b_l^m \tag{6.5}$$

Where ψ^m is the null angle constructed at the m^{th} sensing phase, and α is a multiplying factor. Note that the higher the value of α, higher the null angle is. Now, if the null is wider, chances are more legitimate neighbors fall in nulled region. Node i cannot communicate with its neighbor j if j is in the nulled region of i and vice versa. A higher value of α guarantees a higher probability that the jammer stays in the nulled region until the next sensing period. A very high value of α results in more deactivated links.

In Section 6.2.2.3 we observe that the system performance is a convex function w.r.t. α. Since the jammer's mobility pattern is not completely observable by a node, it should dynamically adjust the value of α. To mitigate this effect, we propose a heuristic that dynamically calculates the value of α based on the observed history of jammer's movements.

6.2.1.3 Heuristic for dynamic α

Algorithm 7 presents a heuristic for adapting the value of α at each sensing period m. Figure 6.7 presents the schema for this procedure. The beam null has been created in the previous sensing period $m-1$. At the m^{th} sensing phase, if the jammer stays inside the nulled region (ψ^{m-1}), then the node successfully avoids the attack. If the jammer is too close to the null border, α is increased. The

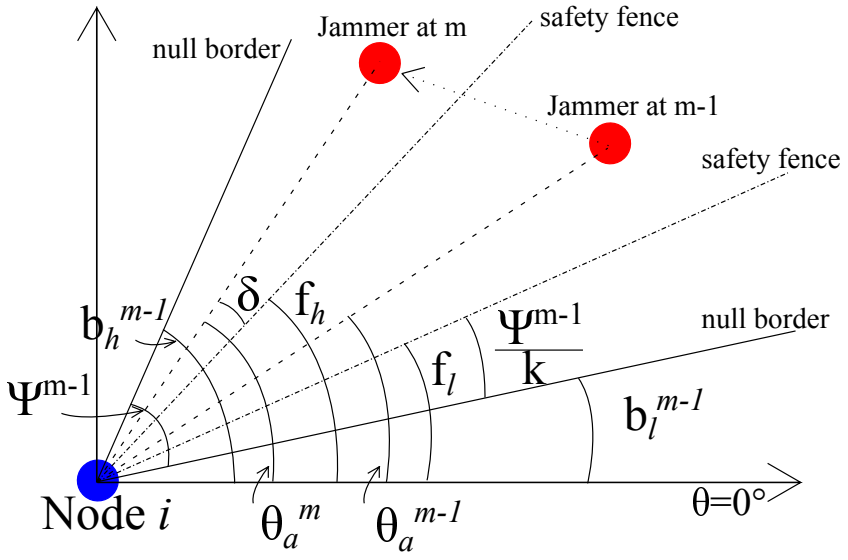


FIGURE 6.7: Schema for adaptive α heuristics

algorithm considers a *safety zone* defined by two fences: f_h and f_l. We consider a factor $k > 2$ which defines how defensive the network is. The safety fence is a ψ^{m-1}/k deviation from the null border towards the center of the null. Larger values of k increase the probability of the jammer being in the safety zone, which consequently decreases α, resulting in a narrower null for the next interval. If the jammer stays inside the safety zone, α is reduced by a factor of $\epsilon \in (0, 1)$. δ is defined as the deviation of the jammer from the safety fence. At the m^{th} sensing phase, if the jammer is observed between the null border and the safety fence, α is increased by a factor of $(1 + \frac{k\delta}{\psi^{m-1}})$. This entails α is doubled if the jammer is at the null border. If the jammer crosses the a border, α is aggressively increased by a multiplying factor of $(1 + (\frac{k\delta}{\psi^{m-1}})^2)$.

6.2.1.4 Calculation of null borders in 3D

From a practical point of view, The 2D framework can be applied to ground and sensor networks under attack by a ground-based jammer. To extend the compatibility of this framework to beam nulling in flying ad hoc networks and 3D mesh scenarios, the framework is generalized by considering 3D distributions of

Algorithm 7: Heuristics for dynamic α

$\psi^{m-1} \leftarrow b_h^{m-1} - b_l^{m-1}$
$f_l \leftarrow b_l^{m-1} + \frac{\psi^{m-1}}{k}$; $f_h \leftarrow b_h^{m-1} - \frac{\psi^{m-1}}{k}$
if $f_l < \theta_a^m < f_h$ **then**
  $\alpha \leftarrow \epsilon\alpha$
else if $\theta_a^m > b_l^{m-1}$ **then**
  $\delta \leftarrow \theta_a^m - f_h$; $\alpha \leftarrow \alpha(1 + (\frac{k\delta}{\psi^{m-1}})^2)$
else if $\theta_a^m < b_l^{m-1}$ **then**
  $\delta \leftarrow f_l - \theta_a^m$; $\alpha \leftarrow \alpha(1 + (\frac{k\delta}{\psi^{m-1}})^2)$
else
  if $\theta_a^m > f_h$ **then**
    $\delta \leftarrow \theta_a^m - f_h$; $\alpha \leftarrow \alpha(1 + \frac{k\delta}{\psi^{m-1}})$
  else
    $\delta \leftarrow f_l - \theta_a^m$; $\alpha \leftarrow \alpha(1 + \frac{k\delta}{\psi^{m-1}})$

nodes and jammer. Therefore, the method of calculating null borders in the 2D framework is extended as follows.

At every sensing phase m, each node i observes the DoA of the jammer or the angle of attack (θ_a^m, ϕ_a^m) with its frame of reference. Let us consider that at the m^{th} sensing phase, node i measures the DoA of jammer as (θ_a^m, ϕ_a^m). Node i has to create a beam null that incorporates the movement of jammer in both θ and ϕ direction during the time interval between sensing phases at m and $m+1$. In 3D space, a beam null is defined by four null borders: two borders in each of θ and ϕ directions. Let us define θ_l^m and θ_h^m as lower and higher null borders respectively in θ direction and ϕ_l^m and ϕ_h^m as lower and higher borders respectively in ϕ direction. Similar to the 2D approach, angular velocity components in θ and ϕ directions are used to predict the movement of the jammer. At each step m, i calculates the angular velocity of the jammer in θ and ϕ directions as v_a^m and u_a^m respectively.

$$v_a^m = \frac{(\theta_a^m - \theta_a^{m-1})}{\tau} \tag{6.6}$$

$$u_a^m = \frac{(\phi_a^m - \phi_a^{m-1})}{\tau} \tag{6.7}$$

Consider $\overline{v_a}$ and $\sigma(v_a)$ as the mean and standard deviation of v_a^m, $(m \in \{1, 2, \dots\})$ respectively. Similarly, $\overline{u_a}$ and $\sigma(u_a)$ are the mean and standard deviation of u_a^m. Node i constructs a beam null using an algorithm that considers the history of jammer's movement. Thus, $(\theta_a^m + \tau\overline{v_a}\ ,\ \phi_a^m + \tau\overline{u_a})$ gives the estimated DoA of the jammer at the $(m+1)^{th}$ phase. Since the actual velocity and direction of the jammer are unknown, the beam null should be wider in case of sudden changes in jammer's direction or velocity change. Change of velocity can be estimated with $\sigma(v_a)$ in θ direction and $\sigma(u_a)$ in ϕ direction. An estimation for the beam null borders in 3D can be calculated as:

$$\theta_h^m = \max(\theta_a^m,\ \theta_a^m + \tau(\overline{v_a} + \alpha\sigma(v_a))) \tag{6.8}$$

$$\theta_l^m = \min(\theta_a^m,\ \theta_a^m + \tau(\overline{v_a} - \alpha\sigma(v_a))) \tag{6.9}$$

$$\psi_\theta^m = \theta_l^m - \theta_l^m \tag{6.10}$$

$$\phi_h^m = \max(\phi_a^m,\ \phi_a^m + \tau(\overline{u_a} + \alpha\sigma(u_a))) \tag{6.11}$$

$$\phi_l^m = \min(\phi_a^m,\ \phi_a^m + \tau(\overline{u_a} - \alpha\sigma(u_a))) \tag{6.12}$$

$$\psi_\phi^m = \phi_l^m - \phi_l^m \tag{6.13}$$

Where ψ_θ^m and ψ_ϕ^m are the null widths constructed at the m^{th} sensing phase in θ and ϕ directions respectively. α is a multiplying factor controlling the influence of randomness in the mobility. As discussed earlier, having a wider null provides higher probability of keeping the jammer inside the beam null at the cost of deactivating more links with legitimate nodes. To keep the beam null optimal based on the history of jammer's DoA observations, a heuristic for dynamic adjustment of α is presented in the next section.

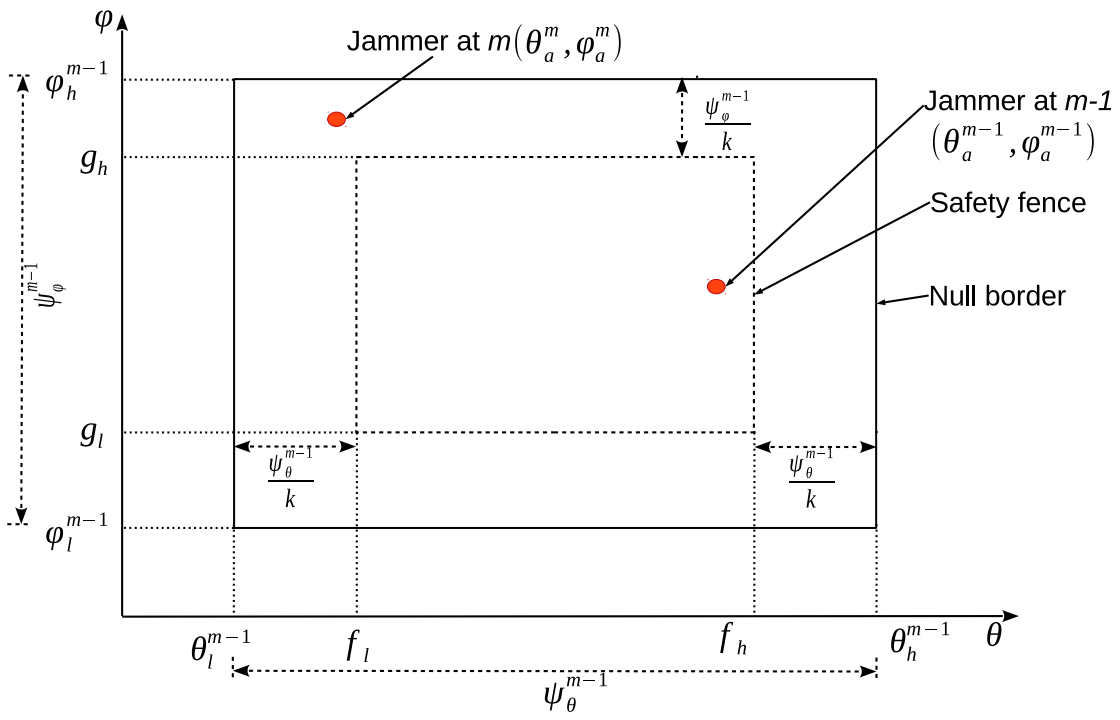


FIGURE 6.8: Schema for adaptive α heuristics for 3D

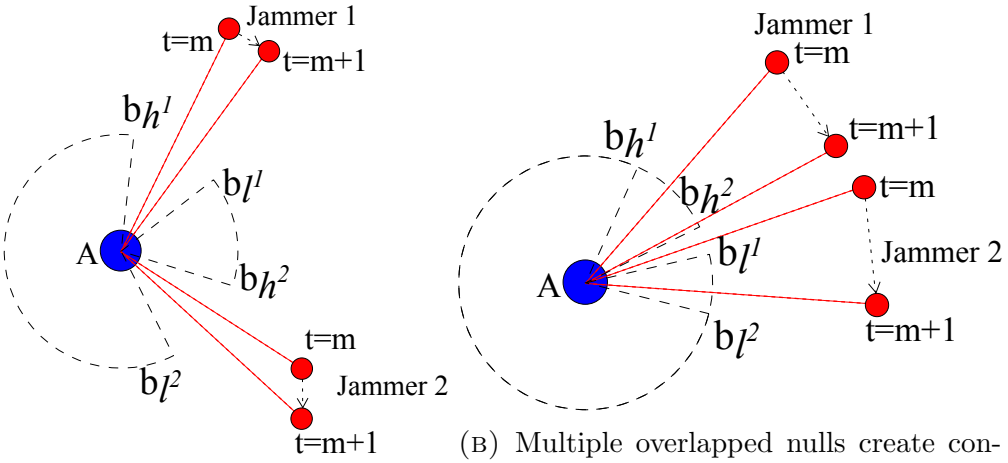
6.2.1.5 Heuristics for dynamic α in 3D

This section demonstrates the concept of dynamically adapting the value of α in 3D space. Unlike the 2D approach, this heuristic in 3D has to consider the DoA of jammer in both directions, as their safety zones are dependent on each other. Figure 6.8 provides a 2D representation of the θ, ϕ space. Algorithm 8 is used at every step m to adjust α based on the observed DoA of jammer at phase m compared to the null created in the phase $m-1$. At phase $m-1$ node i calculates the null borders as $\theta_l^{m-1}, \theta_h^{m-1}, \phi_l^{m-1}$ and ϕ_h^{m-1}. These borders are implemented for the interval between $m-1$ and m. At phase m, DoA of jammer is observed at (θ_a^m, ϕ_a^m). For dynamically changing the value of α, we use a safety zone. The safety zone is bordered by two safety fences in θ direction (f_l, f_h) and two fences in ϕ direction (g_l, g_h). If the current DoA of jammer is within the safety zone, α is decreased by multiplying by a factor $\epsilon \in (0.5, 1)$. If the current location is outside the safety zone, then the deviation of the current position is calculated as γ, which is the maximum value of deviation in both θ and ϕ directions. If $\gamma < 1$ (i.e. the

Algorithm 8: Heuristics for dynamic α in 3D

$\psi_\theta^{m-1} \leftarrow \theta_h^{m-1} - \theta_l^{m-1}$
$f_l \leftarrow \theta_l^{m-1} + \frac{\psi_\theta^{m-1}}{k}$
$f_h \leftarrow \theta_h^{m-1} - \frac{\psi_\theta^{m-1}}{k}$
$\psi_\phi^{m-1} \leftarrow \phi_h^{m-1} - \phi_l^{m-1}$
$g_l \leftarrow \phi_l^{m-1} + \frac{\psi_\phi^{m-1}}{k}$
$g_h \leftarrow \phi_h^{m-1} - \frac{\psi_\phi^{m-1}}{k}$
if $(f_l < \theta_a^m < f_h) \wedge (g_l < \phi_a^m < g_h)$ **then**
    $\alpha \leftarrow \epsilon\alpha$
else
    if $\theta_a^m > f_h$ **then**
        $\delta_\theta \leftarrow \theta_a^m - f_h$
    else if $\theta_a^m < f_l$ **then**
        $\delta_\theta \leftarrow f_l - \theta_a^m$
    else
        $\delta_\theta \leftarrow 0$
    if $\phi_a^m > g_h$ **then**
        $\delta_\phi \leftarrow \phi_a^m - g_h$
    else if $\phi_a^m < g_l$ **then**
        $\delta_\phi \leftarrow g_l - \phi_a^m$
    else
        $\delta_\phi \leftarrow 0$
    $\gamma \leftarrow \max\left(\frac{k\delta_\theta}{\psi_\theta^{m-1}}, \frac{k\delta_\phi}{\psi_\phi^{m-1}}\right)$
    if $\gamma < 1$ **then**
        $\alpha \leftarrow \alpha(1+\gamma)$
    else
        $\alpha \leftarrow \alpha(1+\gamma^2)$

current position is within the safety fence and the null border), α is increased by a small factor. On the other hand if the current DoA of the jammer is outside the null border ($\gamma > 1$), α is increased aggressively by multiplying it with a factor $(1+\gamma^2)$.



(A) Creation of multiple null

(B) Multiple overlapped nulls create consolidated null

FIGURE 6.9: Defense against multipler jammers

6.2.1.6 Defense against multiple jammers

So far we have discussed the calculation of a beam null for a single moving jammer. Each node in a network observes the position of the jammer at discrete sensing intervals (τ). We assume that a node can detect a jammer precisely. In lieu of this assumption, a node can build a model to monitor the trajectory of each jammer within the jamming radius. With an antenna array, a node can adapt its gain pattern to include multiple nulls [160, 161]. A node can create multiple nulls in its modified antenna gain pattern to keep the jammers in the vicinity in null region and communicate with other legitimate nodes that are not in the beam null.

For Each jammer j ($j \in 1, ..., J$) in the vicinity, a node monitors the DoA (θ_j^m, ϕ_j^m) at each sensing period m. The node then use eq. 6.7 to calculate the angular speed of the jammer j w.r.t. the observing node. The beam null borders for the jammer j is calculated using eq. 6.13 at each step m. Figure 6.9a provides an example of defense against multiple jammer. In this case, node A is within jamming radius of 2 jammers. Node a determines beam null borders (b_{l^1}, b_{h^1}) for jammer 1 and (b_{l^2}, b_{h^2}) for jammer 2. Note that, each node maintains separate value of α for each jammer. After observing the position of the jammer at the sensing period $m+1$, the value of α_j is updated using Algorithm 8. It is noteworthy that some

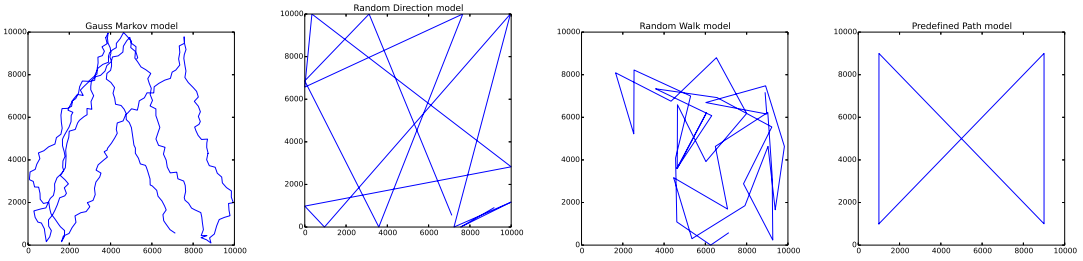
beam nulls can overlap with each other creating a combined beam null as shown in Figure 6.9b.

6.2.2 Results

To evaluate the performance of the proposed beam nulling framework, several simulations are performed. The initial simulations investigate the physical layer behavior of networks employing the proposed framework against jamming attacks. The first of these simulations considers 2D ad hoc networks where the jammer also moves in the same plane that represents the node mobility of ground vehicles. This simulation is further upgraded to emulate similar scenarios for networks and jammer in 3D space. In these simulations, survivability of networks is measured with respect to various physical layer parameters, as well as different mobility models of the jammer. The scope of measurements in then extended to include the behavior of upper layer network protocols. For this purpose, discrete event simulations in ns-3 [162] are performed to monitor the interoperability of the proposed framework with upper network layers. This section defines the parameters and configurations for each simulation, and presents the obtained results through illustrations and discussions.

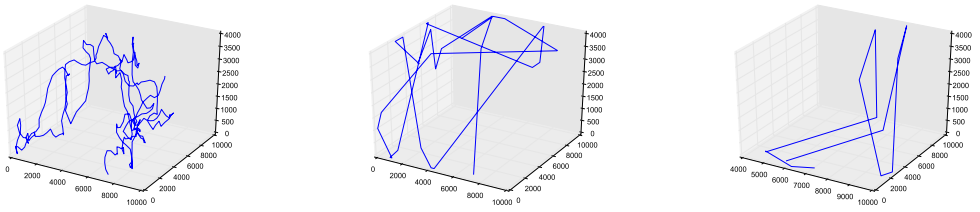
6.2.2.1 Jammer and mobility model

In this work a moving jammer is considered. Different mobility models of the jammer impact differently on a network. A mobility model defines how a node moves or changes its direction with time. The details of the selected models (Random Walk, Random Direction, Gauss-Markov, and a predefined path) can be seen in [163] and [164]. Random-based models are vastly used in the research community but they might not reproduce a realistic movement. Gauss-Markov is a temporal dependency model that can be considered more realistic, where



(A) Gauss Markov (B) Random direction (C) Random Walk (D) Predefined Path

FIGURE 6.10: Time domain sketch of different mobility models in 2D



(A) Gauss Markov (B) Random Direction (C) Random Walk

FIGURE 6.11: Trace of different mobility models in 3D

the velocity and direction are correlated to the previous values, avoiding abrupt changes that occur in the other models. A predefined path is also experimented assuming that a node follows a previously assigned path. Each model has its own influence in the performance of the network. Figures 6.10 and 6.11 illustrate time domain traces of different mobility models in 2D and 3D respectively.

6.2.2.2 Performance metrics

Three performance parameters are defined as follows:

- *Connectivity* is defined as the total number of connected pairs of nodes, which reflects how well connected a network is. More precisely, connectivity of a network is $\frac{1}{2} \times (\sum_{i \in \mathbb{N}} \sum_{j \in \mathbb{N}} connected(i,j))$, where $connected(i,j) = 1$ if there exists at least one path from i to j and 0 otherwise.

- The second parameter is *average number of active links*. We consider a link as the one hop communication between two neighbors. A link may fail if either of the nodes is jammed or falls in the nulled region of the other one.

- The next performance parameter considered is the *average number of islands*. Islands are the subgroups of nodes in a disconnected network where the nodes inside an island are connected. If a network is completely connected, the number of islands is 1. A higher number of islands reflects more disruption in the network.

The simulator monitors the above mentioned metrics at each iteration. It calculates the average of these metrics after the full simulation and records them as the result.

6.2.2.3 Simulation for 2D environment

A customized tick based simulator is developed to measure the performance of the proposed algorithm. Each tick represents the time interval (τ) between two consecutive sensing periods. The default parameters used are listed in Table 6.1. During the sensing phase, at each tick (m), every node checks for the jammer's angular position (θ_a^m). Each node then determines its new beamform according to eq. 6.13 and updates α using Algorithm 7. After the sensing and beamforming phases, communication with neighbors takes place until the time interval (τ) ends, when the same cycle is repeated.

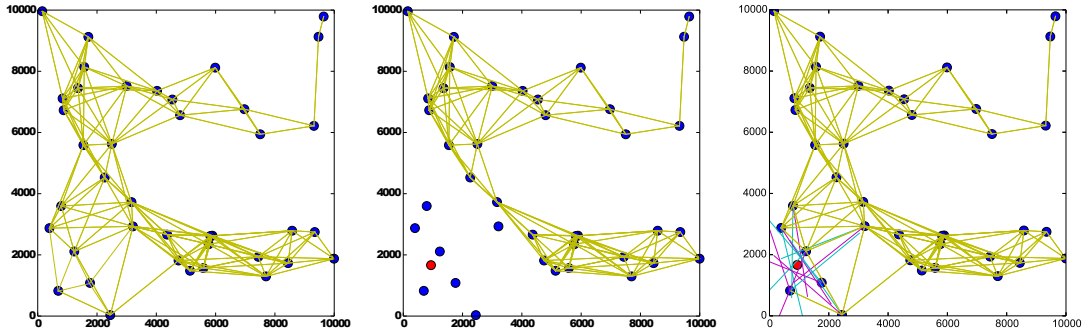
Each simulation generates the position of nodes randomly. The same set of positions is used to measure the performance of the network while varying other parameters. For simplicity, the simulator considers a free space path loss model to calculate the received power. The simulator defines the links between two nodes on each iteration based on the received power from the corresponding neighbor

TABLE 6.1: Default parameters for 2D simulation

Parameters	Symbol	Values
Simulation area		$10,000 \times 10,000\ m^2$
Transmission power	P_t	30 dBm
Received Power cutoff	P_r	-78 dBm
Communication Frequency		2.4 GHz
Communication Radious		3146 m
Initial α	α	2.5
DOA error standard deviation	σ_{doa}	0.05
Beam nulling error standard deviation	σ_{bn}	0.05
Number of nodes simulated	N	100
Sensing interval	τ	50 ms
Simulation Time		500 s
Jammer's mobility model		Random Walk

and interference from the jammer at that instance. If the power received is above the cutoff and neither of the nodes are jammed, the simulator considers the link to be active. The simulator considers a scenario of N nodes scattered randomly in an area of $10,000 \times 10,000\ m^2$. Each node transmits with power of 30 dBm and the average communication radius is calculated as $3146m$.

Figure 6.12 illustrates the advantage of using the proposed framework in the presence of a jammer in 2D environment. Here, 100 nodes are scattered over the geographical area. This snapshot is taken in the middle of a simulation. One hop communication links are represented with yellow lines. Network connectivity of two benchmark scenarios are considered. Figure 6.12a depicts the case of no jamming which leaves the network connected. Figure 6.12b presents network connectivity in the presence of a jammer when nodes use omnidirectional antennas. In this scenario, we can clearly see that many links are deactivated as the nodes are exposed to destructive interference from the jammer. Figure 6.12c demonstrates the effect of employing the proposed framework where the null borders b_l and b_h are represented by cyan and magenta lines respectively. The nodes in the vicinity of the jammer use adaptive beam nulling in order to avoid disruption, and are able to maintain the connectivity with neighboring nodes active. It can also



(A) no jammer, omnidirectional antenna (B) jammer, omnidirectional antenna (C) jammer, adaptive nulling antenna

FIGURE 6.12: Snapshots of simulations

be observed that the nodes which are further from the jammer do not use beam nulling.

The simulator considers the possibility of errors in DoA estimation and beam nulling. As discussed in section 6.2.1.1, we consider the measurement error ($\mathbf{e}_{doa}$) and beam nulling error e_{bn} to be zero mean Gaussian noise. Where $\mathbf{e}_{doa} \sim \mathcal{N}(0, \sigma_{doa})$, and $e_{bn} \sim \mathcal{N}(0, \sigma_{bn})$. Here σ_{doa} and σ_{bn} are standard deviation of error for DoA measurement and beam nulling respectively.

Discrete fixed α : in the initial phase of the simulation, the effect of α on the network's performance is investigated. In this case, the network is simulated without adaptive α, i.e. nodes do not use Algorithm 7. Figure 6.13a presents the simulation results when α is fixed. The x-axis of these plots represent discrete values of α that form the beam null in eq. 6.13. Nine different scenarios are considered: one benchmark scenario with no jamming, and for each mobility model we simulated the network once with omnidirectional antenna, and once with the proposed beam nulling algorithm. The worst case scenario occurs when there is a jammer in the vicinity and the nodes use omnidirectional antenna, consequently the performance is heavily affected by the presence of the jammer. The top benchmark result is obtained similarly to the worst case but with no jammer present, therefore the communications are not affected by any adversary. It can be seen from the results

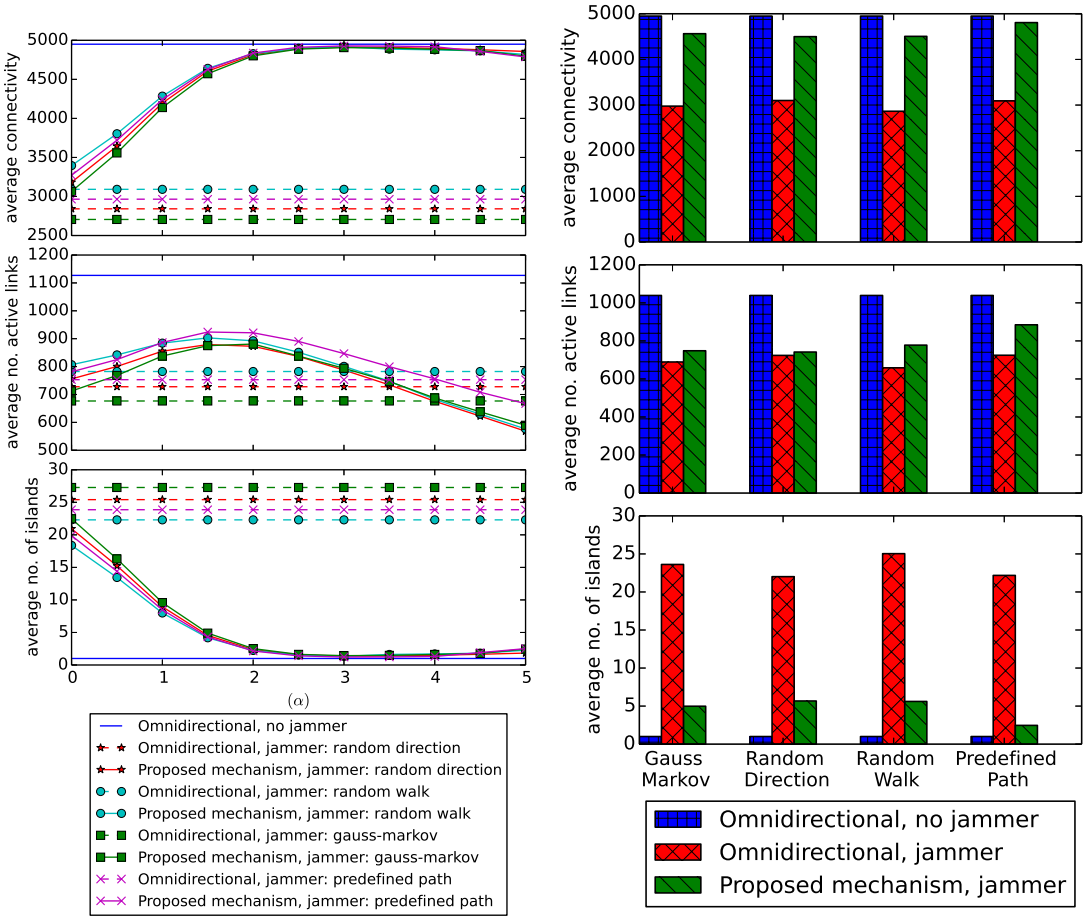
that when there is no jammer, the network is completely connected as the number of islands is 1. For a completely connected network with n nodes, the connectivity value is $\frac{n(n-1)}{2}$. Therefore, in a network of 100 nodes with no jammer, the connectivity is 4950, confirming the simulated result.

When nodes do not use beam nulling, islands are created, resulting in a poor connectivity value. Also it is observed that in the presence of a jammer, adaptive beam nulling significantly improves the overall performance in terms of all the metrics considered. In addition, when a jammer is present and the nodes do not apply beam nulling, the network is heavily affected, and a larger number of islands is created. However, when nodes apply adaptive beam nulling, different trajectory models perform differently with respect to the values of α.

It is noteworthy to mention that for higher values of α, the number of average links may fall below the benchmark case of omnidirectional nodes in the presence of a jammer. This is because a higher value of α creates a wider null that results in deactivation of more links. A node may reduce this shortcoming by sensing the jammer more frequently but this also reduces the data communication window. In addition, it can be observed that as α increases, the average number of islands decreases, while the number of active links begin to deteriorate after a peak. This phenomenon can be interpreted as a rise in congestion.

Another conclusion that can be derived from these results is that a fixed value of α does not guarantee the optimal performance, since the mobility pattern of the jammer is not known to the nodes. A node estimates the jammer's mobility through periodic sensing. Therefore, the value of α must be dynamically updated based on the history of the jammer's movements.

Effect of Jammer's mobility model : four different mobility models for the jammer are considered. Figure 6.13b illustrates the impact of these models on the defending network. It can be seen that the Random Direction and Random



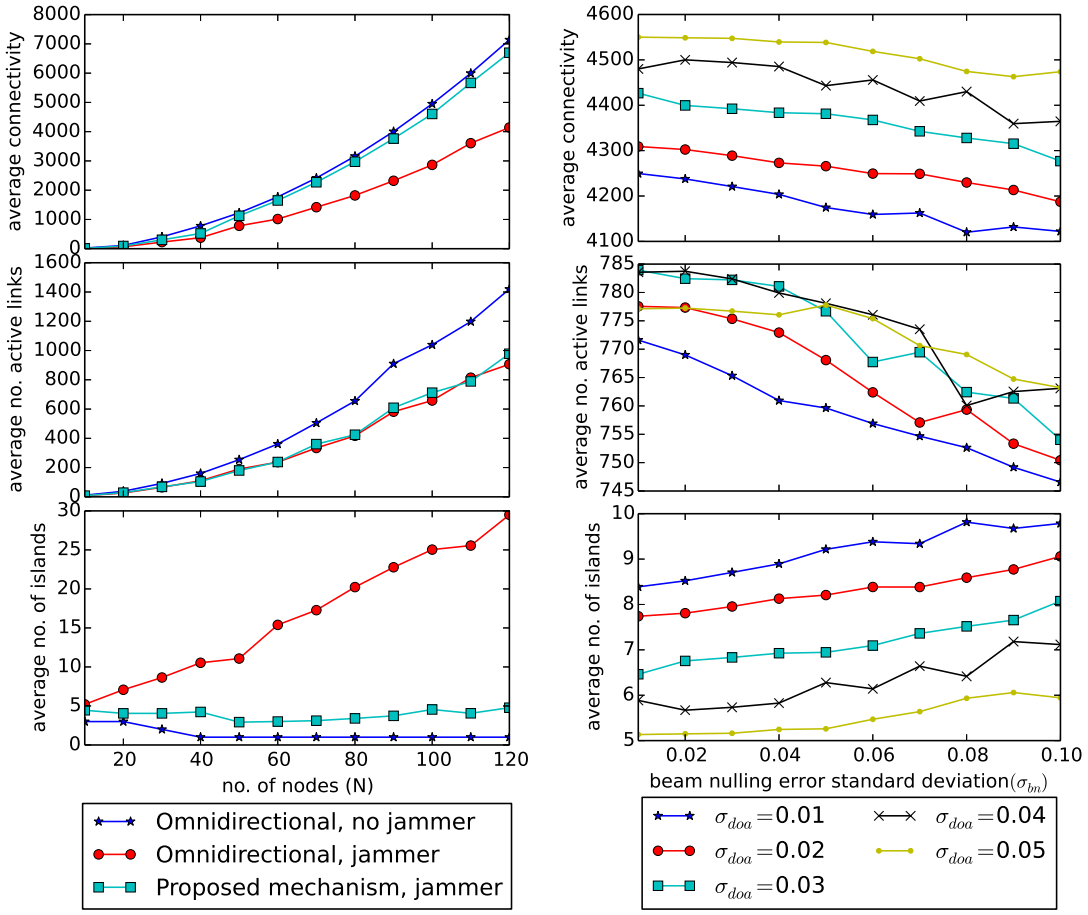
(A) Simulation results with fixed α

(B) Simulation with different trajectory models

FIGURE 6.13: Results for simulation in 2D environment

Walk models adversely affect the performances of the network, since the direction of the jammer undergoes abrupt changes in random intervals. For the predefined path and Gauss-Markov models, the direction and velocity are constant for the majority of the time, which allows the proposed framework to accurately estimate the jammer's movement. It is observed that for 100 nodes, the proposed mechanism achieves an improvement in connectivity of up to 57.27% relative to the omnidirectional case under jamming.

Effect of node density : Figure 6.14a illustrates the effect of varying number of nodes in the network which constitutes a change in node density. It is observed that when a network is not connected, the number of islands increases. As the number



(A) varying number of Nodes (B) varying error rate

FIGURE 6.14: Simulation results varying different simulation parameters in 2D environment

of nodes increases, connectivity is well preserved in the no jamming scenario. The jammer succeeds in disabling more links when the node density is higher. Even though the number of link failures is on a similar level as the worst benchmark of omnidirectional with jamming, connectivity and number of islands demonstrate a better performance. In the benchmark scenario with omnidirectional antennas, the number of islands increases greatly with an increase in the number of nodes, since the density is higher and the attacker has more links in its jamming range. The proposed adaptive beam nulling approach succeeds in keeping the connectivity and number of islands close to the benchmark scenario of no jamming.

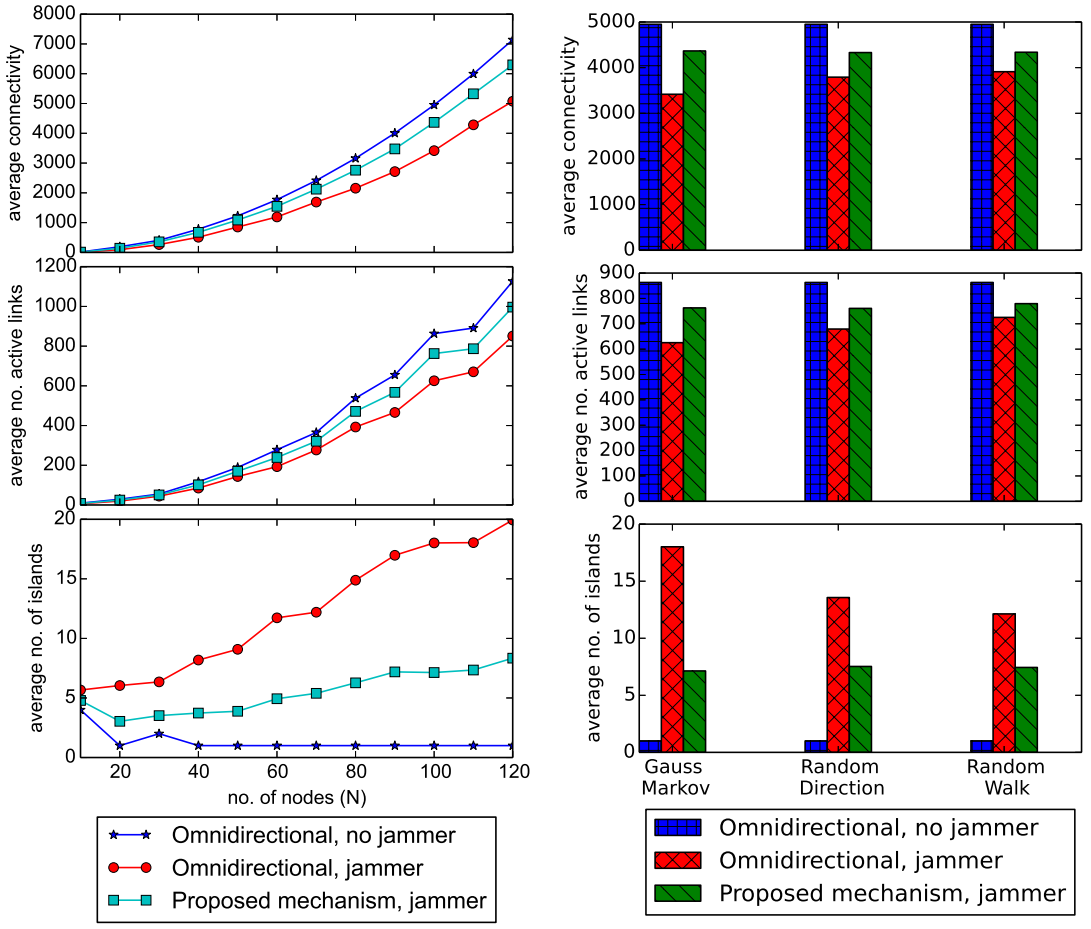
Effect of errors in beam nulling : as discussed earlier, errors are introduced in

the simulator to account for the practical inaccuracies in beam nulling and DoA estimation. The effective beam null border is a random function with the mean of intended border angle and standard deviation of σ_{bn}. Similarly for each node the observed DoA is a random function of mean at the actual DoA and standard deviation of σ_{doa}. Figure 6.14b plots the performance of the network w.r.t. the error in beam nulling. The X-axis is σ_{bn}, while the simulations are repeated with several different values of σ_{doa}. With a σ_{doa} of 0.1 that entails an error of 5.7^o in DoA measurement, the connectivity still remains close to that of the no jamming scenario. The plots reflect that both the errors decrease the network performance significantly as the jammer is not tracked accurately. However, with a higher value of error in measurement, the proposed framework still performs better than the omnidirectional antenna case.

6.2.2.4 Simulation for 3D environment

The simulation is extended to evaluate the performance of our proposed framework in 3D space. The simulation area is a $10 \times 10 \times 4\ km^3$ volume where each node has a communication range of 3 km. Other system related parameters are the same as the 2D simulation discussed earlier. At each tick m, all nodes observe the DoA of the jammer θ_a^m, ϕ_a^m. With the history of the DoA of jammer, a node creates the null as described in section 6.2.1.4. In the next phase, the modified beam is used for data transmission with neighbors. We observe the same system parameters as discussed earlier.

Effect of node density : Figure 6.15a demonstrates the effect of jamming on networks with different node densities. Since the geographical area is fixed, changing number of nodes in the network effectively changes the node density. In this simulation, the jammer moves according to the Gauss Markov mobility model. With lower node density, the network can be partitioned into multiple islands as



(A) Changing node density

(B) Changing jammer's mobility model

FIGURE 6.15: Simulation results for 3D network

seen in the benchmark scenario of no jammer. With increase in node density, the average number of islands is reduced. In the presence of jammer, the network is broken into multiple islands even though the node density is higher. With the proposed framework, networks are able to retain the average number of islands close to the benchmark scenario of no jamming. It is also observed that the proposed framework results in a higher number of active links for the network.

Effect of mobility model of jammer : Figure 6.15b illustrates the performance of the network under jamming with different mobility models. In this simulation, 100 nodes are scattered randomly over the volume. We can clearly see that for the case of omnidirectional antenna with jammer, average connectivity as well as average number of active links are lower for Gauss Markov model compared

to other mobility models. Again, the Gauss Markov model creates more islands than the other mobility models. Thus we can say, in 3D space, a jammer with Gauss Markov mobility model impacts the network most adversely. The proposed framework successfully avoids jamming by creating beam nulls. It can also be observed that although in the omnidirectional case, the jammer with Gauss Markov mobility affects the network performance badly, the proposed framework provides almost the same performance for all the mobility models. So, we can conclude that the framework with dynamic α heuristic is effective in calculating a suitable null width regardless of jammer's mobility model.

6.2.2.5 Simulation with upper layer protocols

To ascertain the effects of the proposed mechanism on the upper network layer protocols, physical layer simulations are extended with network simulations in ns-3 [162], which is a discrete event simulator that provides reliable results when using complex networks with multiple protocol stacks. The simulations are focused towards the interoperability of the proposed framework with two ad hoc routing protocols, namely AODV and DSDV. For both routing protocols the IEEE 802.11b MAC is used. Ad hoc On-demand Distance Vector (AODV) is a reactive routing protocol with some active elements. In this scheme, the routes are discovered only when needed, but they are maintained for as long as possible. This can cause delay when there is data ready to be transferred by a node, but no route is stored in its routing table [165]. Destination-Sequenced Distance Vector (DSDV) is a proactive protocol, meaning it will regularly update the routing table, even when there is no data to be transmitted. DSDV requires new sequence numbers before the topology can converge again, making it's implementation in highly mobile networks undesirable [166].

TABLE 6.2: Simulation parameters for ns-3

Parameter	Value
Number of nodes	100
Tick interval	50 ms
Simulation time	500 s
Transport layer protocol	TCP
Dimension	$10,000 \times 10,000\ m^2$
Number of sources	10
Number of destinations	10
MAC protocol	IEEE 802.11b
Receiver Sensitivity	-78 dBm
Propagation loss model	Friis free space propagation
Data rate	1 Mbps

TABLE 6.3: Application layer parameters

Application	Bytes generated	Probability
Text	10000	0.6
Image	500000	0.3
Video	5000000	0.1

To simulate the proposed mechanism, a proof of concept antenna model [167] is used, which contains a few parameters: beamwidth, gain inside the beamwidth, gain outside the beamwidth, and orientation. In our case, the beamwidth corresponds to the nulled region, and the gain inside it is set to -60 dB, the gain outside is 0 db, and the orientation is defined as the direction towards the jammer. The traffic in the application layer is generated by 10 random source nodes and received by 10 random destination nodes. Details of the simulation parameters are presented in Table 6.2. Three different applications are used to send data at different rates. The amount of data to be transferred is randomly chosen by each source according to the probabilities shown in Table 6.3. Tables 6.4 and 6.5 provide the default parameters used for AODV and DSDV protocols.

The simulator provides four performance metrics: *packet delivery ratio (PDR), mean hop count, mean delay* and *bytes received*. PDR is the ratio between the number of packets received by the destination to the number of packets sent by the source. Mean hop count is the average number of hops taken by the packets in

TABLE 6.4: Parameters for simulating AODV

Parameters	Values
Hello interval	1 s
RREQ retries	2
RREQ rate limit	10 per second
RERR rate limit	10 per second
Node traversal time	40 ms
Next hop wait	50 ms
Active route timeout	3 s
Net diameter	35
Max queue length	64 packets
Max queue time	30 s
Allowed hello loss	2
Enable hello	TRUE
Enable broadcast	TRUE

TABLE 6.5: Parameters for simulation of DSDV

Parameters	Value
Periodic update interval	15 s
Max queue length	500 packets
Max queue time	30 s
Max queue per destination	5packets

the simulation (including control packets) to reach their destinations. Mean delay is the average time taken for the packets (including control packets) to reach their destinations. Bytes received is the amount of bytes received by the destination nodes in the application layer. Figures 6.16a and 6.16b illustrate the performance of a network under jamming using AODV and DSDV respectively. Three different mobility models are explored for the jammer: Gauss-Markov, Random Direction, and Random Walk.

The upper subplot in both Figures 6.16a and 6.16b represent PDR value w.r.t different mobility models in the simulation. For both of the routing protocols, the proposed framework ensures enhanced PDR. It can be concluded that regardless of the routing protocol or the mobility pattern, the proposed framework is able to provide enhancement in the performance compared to the case of omnidirectional network under jamming.

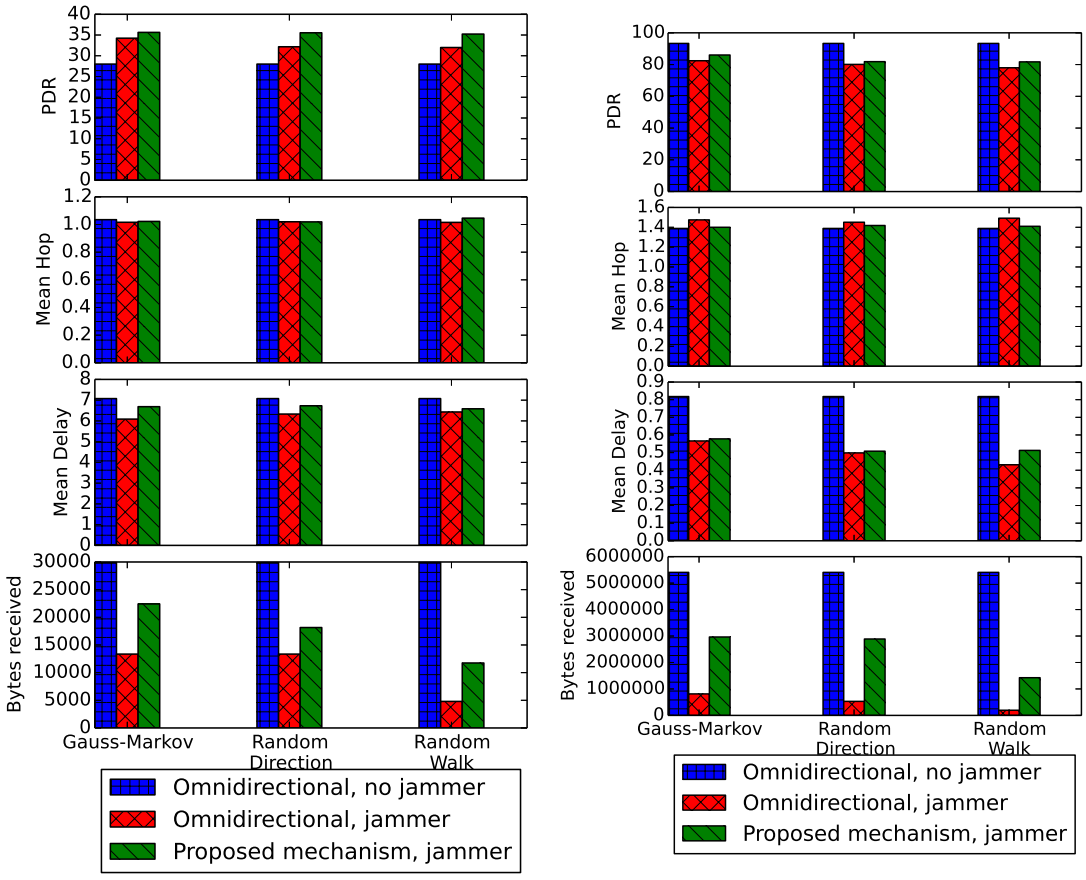
The mean hop count plot shows a negligible difference for AODV, but in DSDV it is evident that the proposed framework improves this metric by keeping links active even when their corresponding nodes are inside the jammed region. In the presence of the jammer the hop count is higher than in the other cases.

For DSDV as in Figure 6.16b, it can be seen that the mean delay in our proposed framework is lower compared to the benchmark scenario of omnidirectional without jammer. As beam nulling is used, nodes experience less interference from the neighbors, reducing the waiting time for packets in the queue, since there is less collisions due to medium access conflicts. With the reduced waiting time, the packets are transferred to the destination faster. This means that the proposed framework not only retains links in the jammed region but also reduces congestion on the links outside of the jammed region.

The last subplot illustrates the amount of data received by all destination nodes. It is observed that DSDV outperforms AODV by a large margin for all mobility models. This is partly due to the resolution of physical layer simulations, which cause the loss of some AODV messages during the network simulation. DSDV on the other hand is a proactive protocol, keeping the routes updated as link failures are detected. This characteristic plays an advantage and gives DSDV the better performance in the simulations.

6.2.2.6 Simulation with multiple jammers

We simulated the network with multiple jammers to illustrate the behavior of the adaptive beam nulling method as described in Section 6.3.1.6. The simulated network consists of 100 nodes, the rest of the parameters are kept same as before as listed in Table 6.1. In Figure 6.17 we plot the simulation results. Simulations are performed for different number of jammers in between 0 and 5, where 0 represents the case of no jammer as a bench mark of best case scenario.



(A) AODV

(B) DSDV

FIGURE 6.16: Simulation results from ns-3 simulation

Results show that using the adaptive beam nulling improves the network connectivity when compared with the same case with omnidirectional antenna. In comparison with the benchmark scenario of no jammer, the adaptive beam nulling approach shows a decrease of 39.94%, while using omnidirectional antenna decreased to 97.73%.

The number of active links and the number of islands also show improvements. The average number of active links decreased 72.05% with the proposed mechanism and 91.3% when no protection was used. Even with the apparent large decrease, the number of islands with the proposed beam nulling is 5.92 times lower than the omnidirectional case, demonstrating the feasibility of the proposed method for defense against multiple jammers.

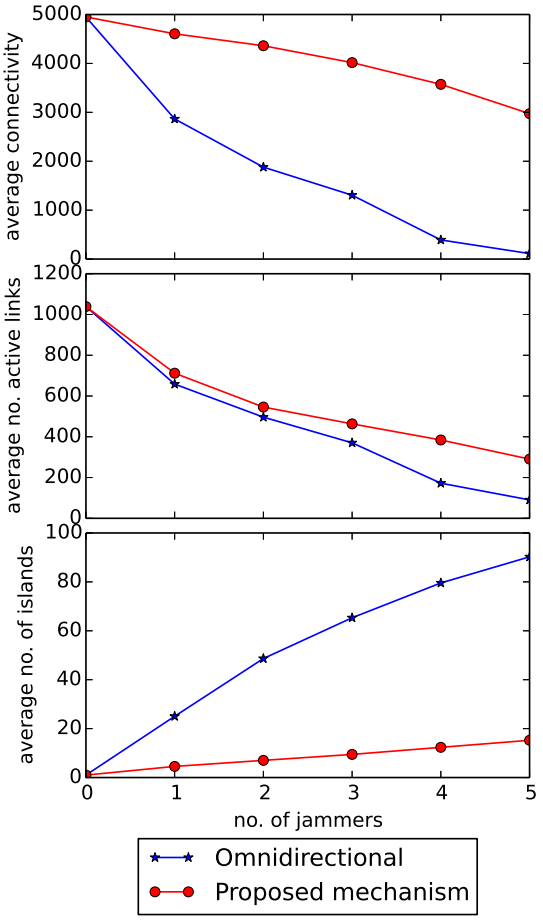


FIGURE 6.17: Simulation results for multiple jammer

6.3 Optimized beamnull with Kalman filter

This section describes the proposed technique of observing jammer's movement, predicting its next movement and obtaining an optimal beam null. Furthermore, we extend our study to derive optimal beam nulls in presence of multiple jammers.

6.3.1 Methodology

6.3.1.1 Problem statement

Figure 6.18 provides an illustration of the effect of beamnulling against a moving jammer. The picture emphasizes the effect of null region size of node a in between

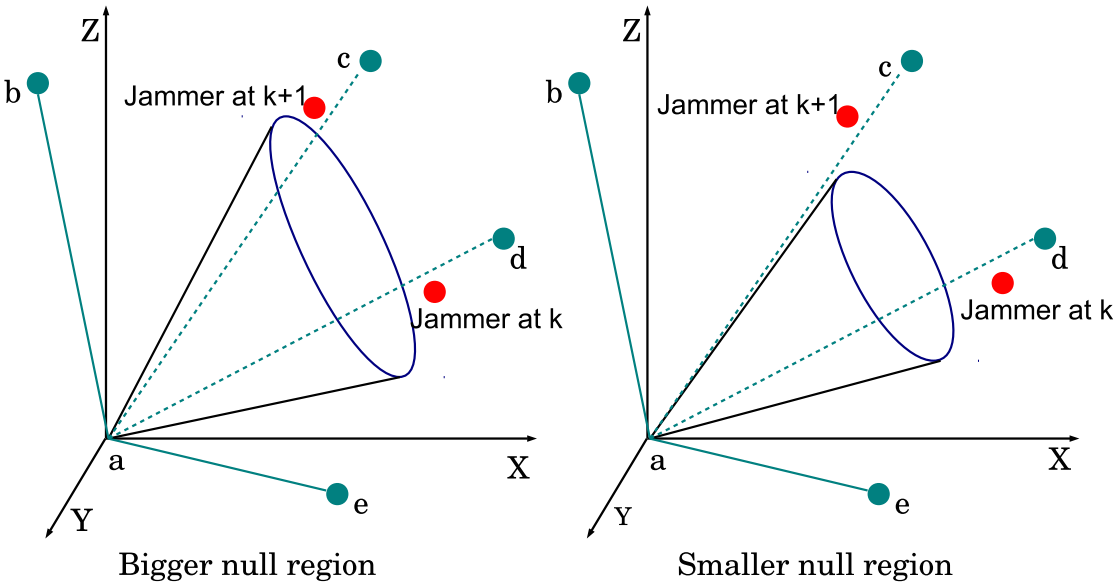


FIGURE 6.18: Trade off of having wider null region

sensing intervals k and $k+1$. The one hop links for node a with its neighbors b, c, d and e are depicted here. Node a observes the DoA of the jammer at every sensing period $k \in [0, 1, \dots]$. As the observation is not continuous, a takes into account the movement of jammer in between the sensing periods. By learning from the history of the position of the jammer, a node can predict the probable trajectory of the attacker at step $k+1$. If movement pattern of the jammer is random, the prediction accuracy decreases. Therefore a buffer region should be considered which will guarantee to keep the movement of the jammer within the buffer zone. This buffer zone can be used to create the beam null.

Again, Figure 6.18 presents two scenarios of the beam nulling. If a uses a bigger null, the probability of the jammer movement being inside the null increases. Having a bigger null increases the number of neighbors to be shadowed in the null, which in turn causes the link with the shadowed neighbors to be disconnected. With a bigger null, a can maintain links with b and e whereas links to c and d fail. With a smaller null as depicted in the figure, a can preserve link c. However, as the jammer moves to the position in step $k+1$, the jammer falls outside of the beam null used by a. As soon as a is exposed to the jammer, a would experience

jamming that results in failure in all links of a. The trade-off for widening the beam null to cover the probable movement of the jammer with higher confidence comes at a cost of disabling some unaffected links. Hence, the goal of this chapter is to derive an optimization technique that considers the cost and benefits of beam null and finds out the optimal beam null region.

6.3.1.2 Tracking movement of a jammer with noisy observation

The Kalman filter provides an estimation of the position of the jammer when there is an error in obtaining the jammer's position. After obtaining the history of possible jammer's position, the node obtains the next possible position of the jammer using multivariate time series analysis. The Kalman filter is a recursive equation that aims at minimizing the mean-square estimation error of a random variable $\mathbf{x}$. It assumes that a random process to be estimated can be modeled in the form of

$$\mathbf{x_{k+1}} = \mathbf{F_k x_k} + \mathbf{w_k} \tag{6.14}$$

Measurements of this process occur at discrete time intervals. The filter assumes a linear relationship between the observation and the actual state of the process

$$\mathbf{z_k} = \mathbf{H_k x_k} + \mathbf{v_k} \tag{6.15}$$

Where,

$\mathbf{x}_k$ = $(n \times 1)$ state vector at time t_k

$\mathbf{F}_k$ = $(n \times n)$ state transition matrix relating x_k to x_{k+1}

$\mathbf{w}_k$ = $(n \times 1)$ input white noise with known covariance

$\mathbf{z}_k$ = $(m \times 1)$ observation or measurement at time t_k

$\mathbf{H}_k$ = $(m \times n)$ observation matrix giving the noiseless connection between the measurement and the state vector

$\mathbf{v}_k$ = $(m \times 1)$ white sequence measurement error with known covariance

The filter assumes that $\mathbf{F_k}, \mathbf{H_k}$, and the covariance matrix describing $\mathbf{w_k}, \mathbf{v_k}$ are known. The covariance matrices for the $\mathbf{w_k}$ and $\mathbf{v_k}$ are given by

$$\mathbf{E}[\mathbf{w_k}\mathbf{w_i^T}] = \begin{cases} \mathbf{Q}_k & , \quad i = k \\ 0 & , \quad i \neq k \end{cases} \tag{6.16}$$

$$\mathbf{E}[\mathbf{v_k}\mathbf{v_i^T}] = \begin{cases} \mathbf{R}_k & , \quad i = k \\ 0 & , \quad i \neq k \end{cases} \tag{6.17}$$

$$\mathbf{E}[\mathbf{w_k}\mathbf{v_i^T}] = 0 \quad , \forall k, \forall i \tag{6.18}$$

Figure 6.19 provides a representation of the Kalman filter process [2]. It starts with an initial or apriori estimate about the first observation and its covariance. At every step k, it takes measurement $\mathbf{z_k}$ and updates the estimated state ($\widehat{\mathbf{x_k}}$) of the actual process. The covariance of the estimated state ($\mathbf{P_k}$) is also updated. Then it predicts the state of the actual process on the next step ($\widehat{\mathbf{x_{k+1^-}}}$) and the covariance of the predicted next step ($\mathbf{P_{k+1^-}}$). Then it updates the gain of the filter $\mathbf{K_k}$ and waits for the next measurement.

Now, for our system, the actual state of the jammer ($\mathbf{x_k}$) consists of four variables: $\theta, \dot{\theta}, \phi, \dot{\phi}$. Here $\dot{\theta}$ and $\dot{\phi}$ are velocity in θ and ϕ directions respectively. We can

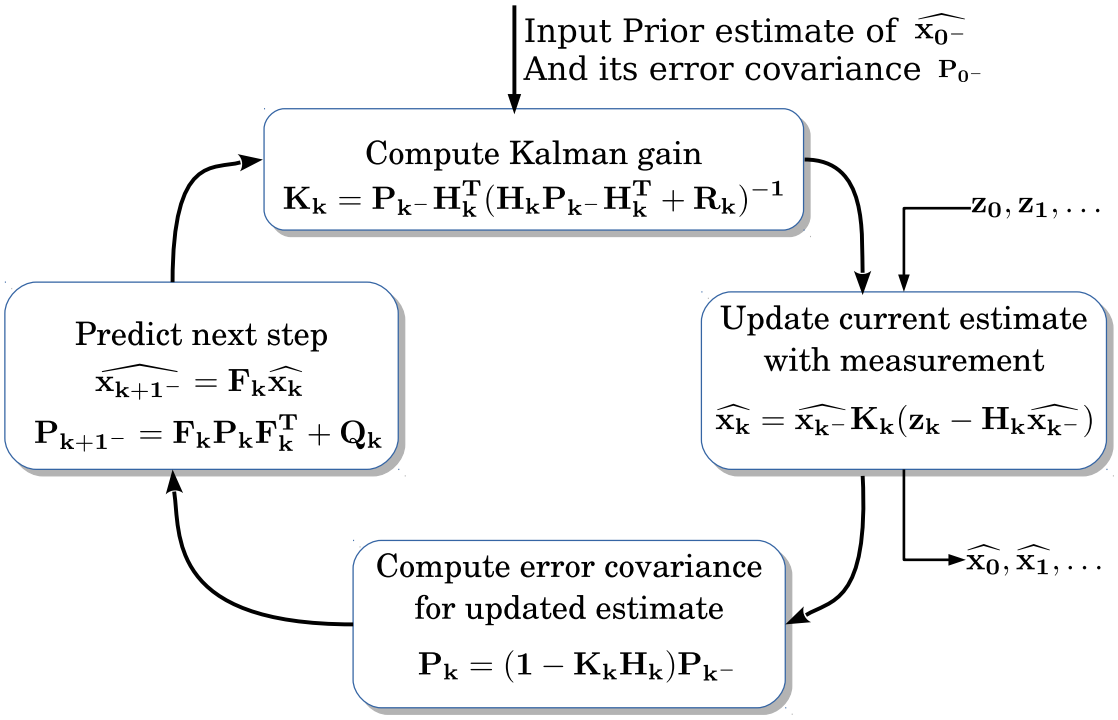


FIGURE 6.19: Kalman filter iteration [2]

write eq. 6.14 as

$$\begin{bmatrix} \theta(k+1) \\ \dot{\theta}(k+1) \\ \phi(k+1) \\ \dot{\phi}(k+1) \end{bmatrix} = \begin{bmatrix} 1 & \tau & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \tau \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \theta(k) \\ \dot{\theta}(k) \\ \phi(k) \\ \dot{\phi}(k) \end{bmatrix} + \mathbf{w_k} \tag{6.19}$$

A node can observe only the position of the jammer in terms of θ and ϕ. Then eq. 6.15 can be written as

$$\begin{bmatrix} z_\theta(k+1) \\ z_\phi(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \theta(k) \\ \dot{\theta}(k) \\ \phi(k) \\ \dot{\phi}(k) \end{bmatrix} + \mathbf{v_k} \tag{6.20}$$

As the error in DoA measurement or noise in the process follow Gaussian distribution, we can consider,

$$\mathbf{w_k} \sim \mathcal{N}(0, \mathbf{Q}) \tag{6.21}$$

$$\mathbf{v_k} \sim \mathcal{N}(0, \mathbf{R}) \tag{6.22}$$

where $\mathbf{v_k}$ is the DoA estimation error while $\mathbf{w_k}$ is the error or displacement of the jammer from it's intended position.

6.3.1.3 Constructing beam null

At each step k, a node observes the position of jammer in terms of θ and ϕ. This observation is fed to the Kalman estimator which determines the estimated current position of the jammer ($\widehat{\mathbf{x_k}}$) and predicts position of the jammer at next step ($\widehat{\mathbf{x_{k+1^-}}}$). We construct two circles $\bigcirc_A$ and $\bigcirc_B$ whose centers are at the current estimate and predicted position respectively.

$$\begin{bmatrix} \theta_A \\ \phi_A \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \widehat{\mathbf{x_k}} \tag{6.23}$$

$$\begin{bmatrix} \theta_B \\ \phi_B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \widehat{\mathbf{x_{k+1^-}}} \tag{6.24}$$

Two confidence regions are determined that enforces certain confidence level for the estimation process. Having a bigger diameter for the confidence circles will result in a greater probability that the jammer is inside the circle. We consider the diameter of the circles to be s times the standard deviation of the estimated position and the predicted position. The filter also provides two covariance matrices: covariance for the current position estimation ($\mathbf{P_k}$) and covariance for predicted position in next

step ($\mathbf{P_{k+1^-}}$). $\mathbf{P_k}$ contains $cov_k(\theta,\theta)$ and $cov_k(\phi,\phi)$. $\mathbf{P_{k+1^-}}$ contains $cov_{k+1^-}(\theta,\theta)$ and $cov_{k+1^-}(\phi,\phi)$. The radii for $\bigcirc_A$ and $\bigcirc_B$ are respectively,

$$r_A = \frac{s}{2}\sqrt{\max(cov_k(\theta,\theta), cov_k(\phi,\phi))} \tag{6.25}$$

$$r_B = \frac{s}{2}\sqrt{\max(cov_{k+1^-}(\theta,\theta), cov_{k+1^-}(\phi,\phi))} \tag{6.26}$$

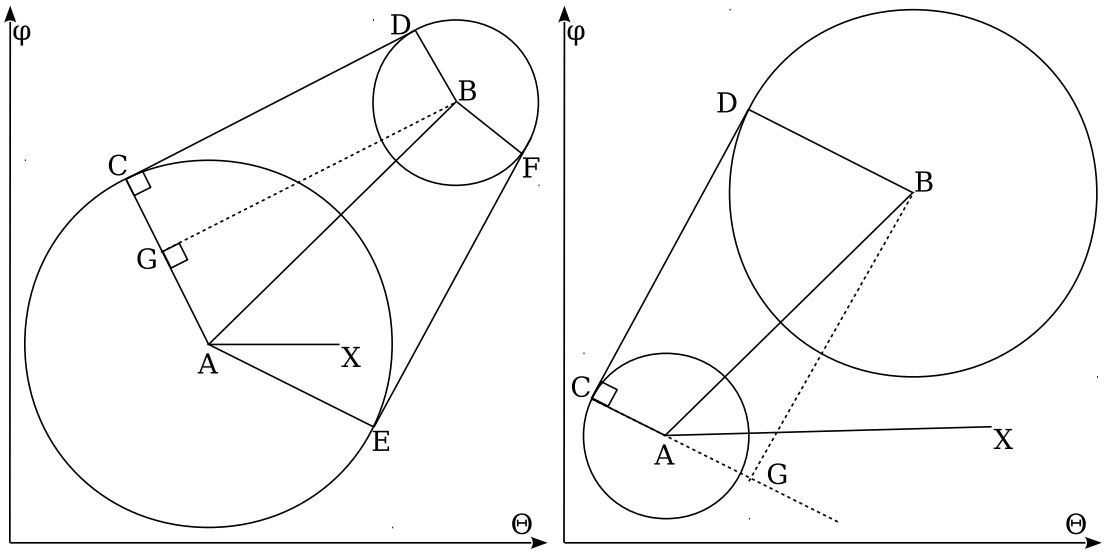
The beam null contains two circles and the region where the jammer may be in between two measurement updates. It is estimated that jammer is inside $\bigcirc_A$ at step k and predicted to be inside $\bigcirc_B$ at $k+1$. If the jammer moves straight in between the two measurement interval then it can only move in the area covered by the two circles and their outer tangents. If one circle cover the entire region (i.e. one circle stays inside the other) then the beam null will only be the bigger circle. The condition for this is as follows:

$$\max(r_A, r_B) > \min(r_A, r_B) + \sqrt{(\theta_A - \theta_B)^2 + (\phi_A - \phi_B)^2} \tag{6.27}$$

If the above condition is not valid then the null area is determined by calculating the common outer tangents.

Determining the outer tangents : let the center points of two circles $\bigcirc_A$ and $\bigcirc_B$ be $A(\theta_A, \phi_A)$ and $B(\theta_B, \phi_B)$ respectively. The radii for these two circles are r_A and r_B respectively. Let us have a look at representation of θ and ϕ on a 2D plane as illustrated in Figure 6.20a. From the previous section we have calculated $\theta_A, \phi_A, r_A, \theta_B, \phi_B$ and r_B. Now we are interested in determining the outer tangents that connect these two circles.

Let us consider the case of Figure 6.20a where $r_A > r_B$. Let us consider the tangents are CD and EF whose coordinates are unknown at this point. A tangent



(A) $r_A > r_B$ (B) $r_A < r_B$

FIGURE 6.20: Schema to obtain common outer tangents

of a circle is always perpendicular to the line that connects the touching point and the center. Thus, $CD \perp AC$ and $CD \perp BD$. Which entails that $AC \parallel BD$.

Now, lets consider a point G on line AC such that length of $CG = r_B$. As $CD \perp AC$, $CD \perp BD$, and $CG = BD$, quadruple $GBDC$ is a rectangle. Thus, $GB \perp CG$. This entails, $GB \perp AG$. Now we can calculate:

$$\angle GAB = \cos^{-1} \frac{AG}{AB} = \cos^{-1} \frac{r_A - r_B}{\sqrt{(\theta_A - \theta_B)^2 + (\phi_A - \phi_B)^2}} \tag{6.28}$$

Let us draw a line AX that is parallel to θ axis. We can calculate:

$$\angle XAB = \tan^{-1} \frac{\phi_B - \phi_A}{\theta_B - \theta_A} \tag{6.29}$$

Note that tan^{-1} provides same angle for first and third quadrant. So, we checked the sign of numerator and the denominator and then corrected the formula during simulation. If the target point is in third or fourth quadrant relative to the observer, then π should be added to the tan^{-1} angle.

As $AC \parallel BD$, both of these lines are making the same angle with θ axis. The angle is $\angle GAB + \angle XAB$. Thus we can determine the position of C and D as:

$$\theta_C = \theta_A + r_A \cos(\angle GAB + \angle XAB) \tag{6.30}$$

$$\phi_C = \phi_A + r_A \sin(\angle GAB + \angle XAB) \tag{6.31}$$

$$\theta_D = \theta_B + r_B \cos(\angle GAB + \angle XAB) \tag{6.32}$$

$$\phi_D = \phi_B + r_B \sin(\angle GAB + \angle XAB) \tag{6.33}$$

We know that the lines connecting the center and two outer common tangents make same angle with the line connecting the centers of the circles. Thus, $\angle EAB = \angle GAB$. So, $\angle XAE = \angle XAB - \angle GAB$. We can determine the positions for E and F as:

$$\theta_E = \theta_A + r_A \cos(\angle XAB - \angle GAB) \tag{6.34}$$

$$\phi_E = \phi_A + r_A \sin(\angle XAB - \angle GAB) \tag{6.35}$$

$$\theta_F = \theta_B + r_B \cos(\angle XAB - \angle GAB) \tag{6.36}$$

$$\phi_F = \phi_B + r_B \sin(\angle XAB - \angle GAB) \tag{6.37}$$

It can be easily proven that these equations also hold true for the case of $r_A < r_B$ as demonstrated in Figure 6.20b. Regardless of radii of the circles, we can use the above equation.

Determining if a node is inside beam null : the node needs to determine how many links would be broken due to the newly created null region. If a node j has DoA of (θ_j, ϕ_j) relative to node i, then j would be inside beam null if the DoA is inside either circle A, or circle B or inside the quadruple $CDEF$. Figure 6.21 depicts a 2D representation of θ, ϕ.

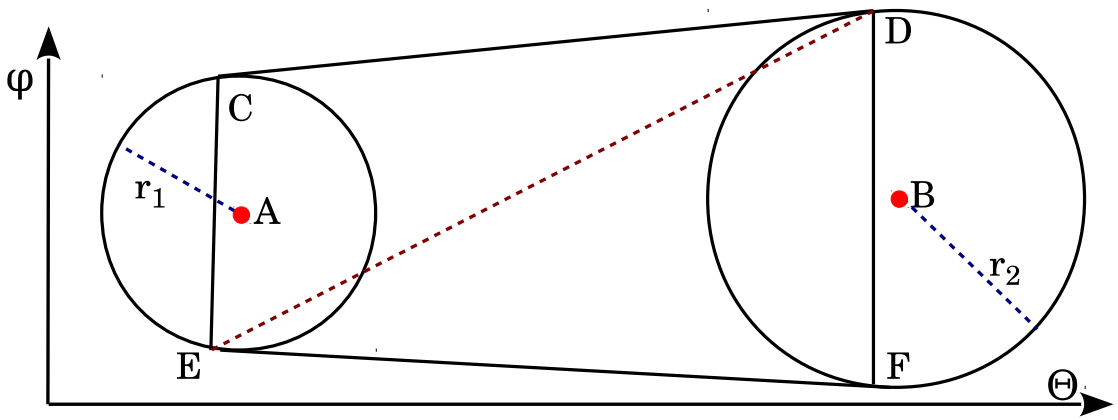


FIGURE 6.21: Cross section of beam null in (θ, ϕ)

Total area covered by the null is

$$Area(\bigcirc_A) \cup Area(\bigcirc_B) \cup Area(\square_{CDFE})$$

Now, let us consider the case of the quadruple $CDFE$. As two sides of this quadruple are outer tangent of two circles, we can divide the area into two non overlapping triangles: $\triangle_{CDE}$ and $\triangle_{DEF}$. So, the condition whether DoA of j is inside the null is:

$$\varrho_j = \varrho_{\bigcirc_A} \vee \varrho_{\bigcirc_B} \vee \varrho_{\triangle_{CDE}} \vee \varrho_{\triangle_{DEF}} \tag{6.38}$$

Where $\varrho_{\bigcirc_A}, \varrho_{\bigcirc_B}, \varrho_{\triangle_{CDE}}, \varrho_{\triangle_{DEF}}$ are conditions for being inside circle A, circle B, triangle CDE and triangle DEF respectively. Note that if one circle is inside another circle as determined in eq. 6.27, we have to check only for the bigger circle. Conditions for being inside circle A and B are:

$$\varrho_{\bigcirc_A} = \sqrt{(\theta_j - \theta_A)^2 + (\phi_j - \phi_A)^2} < r_A \tag{6.39}$$

$$\varrho_{\bigcirc_B} = \sqrt{(\theta_j - \theta_B)^2 + (\phi_j - \phi_B)^2} < r_B \tag{6.40}$$

Let us again look at the θ, ϕ representation on a 2D plane as in Figure 6.21. A point $j(\theta_j, \phi_j)$ is inside triangle CDE if the area of triangle CDE is same as the sum of area of triangles jDE, CjE, and CDj. The area of a triangle CDE can

be calculated as:

$$\mathfrak{A}(\triangle_{CDE}) = \left| \frac{\theta_C(\phi_D - \phi_E) + \theta_D(\phi_E - \phi_C) + \theta_E(\phi_C - \phi_D)}{2} \right|$$

The condition to check for the DoA of j inside triangles are:

$$\varrho_{\triangle_{CDE}} = \begin{cases} 1 & if\ \mathfrak{A}(\triangle_{CDE}) = \mathfrak{A}(\triangle_{jDE}) + \mathfrak{A}(\triangle_{CjE}) + \mathfrak{A}(\triangle_{CDj}) \\ 0 & otherwisw \end{cases}$$

$$\varrho_{\triangle_{DEF}} = \begin{cases} 1 & if\ \mathfrak{A}(\triangle_{DEF}) = \mathfrak{A}(\triangle_{jEF}) + \mathfrak{A}(\triangle_{DjF}) + \mathfrak{A}(\triangle_{DEj}) \\ 0 & otherwisw \end{cases}$$

6.3.1.4 Optimization goal

Let $\mathbb{N}$ be the set of nodes in a 3D mesh network. Let $j \in \mathbb{N}$ be an one hop neighbor of $i \in \mathbb{N}$. With a known beam null, i can assess the probability of the failure of link with j. Considering that j is not jammed, we can determine the probability that link ij fails as

$$\mathbb{P}(\text{ij fails}) = \begin{cases} 1 & if \quad \varrho_j = True \\ \mathbb{P}(\text{i is jammed}) & otherwise \end{cases} \tag{6.41}$$

$$= \varrho_j + (1 - \varrho_j)\mathbb{P}(\text{node i is jammed}) \tag{6.42}$$

The probability of a node successfully avoiding jamming is same as the probability that the jammer stays within the null during next transmission interval. As the error in the DoA estimation model is a normal distribution, we can say that probability of successful estimation would closely follow Chebyshev's inequality. In that case, if node i uses s_i standard deviation in eq. 6.25 and eq. 6.26 for

calculating circle diameters,

$$\mathbb{P}(\text{jammer in the estimated region}) \approx 1 - \frac{1}{s_i^2} \tag{6.43}$$

For the optimization purpose, we can consider,

$$\mathbb{P}(\text{node i being jammed}) = \frac{1}{s_i^2} \tag{6.44}$$

Thus, we can write,

$$\mathbb{P}(\text{ij fails}) = \varrho_j + \frac{(1-\varrho_j)}{s_i^2}) \tag{6.45}$$

In 3D mesh networks every link has a different importance level in the network. For example if a link is relaying data from many nodes or a link is transmitting crucial data, it can be assigned higher weight. It is for the best interest of the network that these links are safe guarded from failure. Let $w_{ij};\ i,j \in \mathbb{N}$ denote the weight for a link between i and j. If all links are equally important for a network then $w_{ij} = 1; \forall i,j$. Thus, the optimization problem becomes:

$$maximize \quad \sum_{j \in \mathbb{N}} w_{i,j} \left(\varrho_j + \frac{(1-\varrho_j)}{s_i^2}) \right) \tag{6.46}$$

The lower value of s_i reduces the beam null region that in effect reduces number of deactivated links. But it comes with a cost that there is a higher probability of i being jammed that, in effect, deactivates all links of i. Again, a very high value of s_i increases the number of deactivated links. The maximization problem stated above is a convex optimization problem that computes optimal s_i at every step.

6.3.1.5 Algorithm

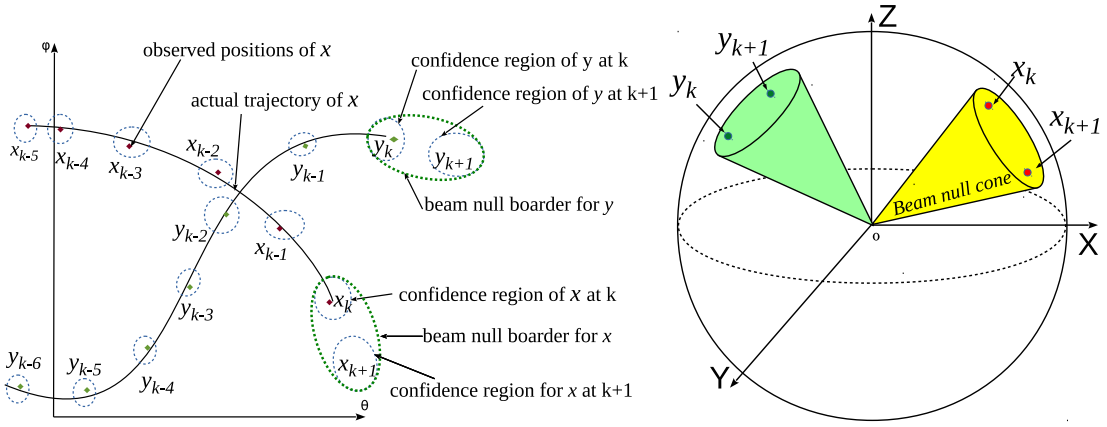
Each node $i \in \mathbb{N}$ follows Algorithm 9 at each step k to create the beam null. At first i observes the position of the jammer $z_\theta(k), z_\phi(k)$. If it is being jammed and there is not enough data to predict the possible trajectory of the jammer, i creates a beam null cone centering $z_\theta(k), z_\phi(k)$ using a threshold value r_{th} as the radius of the beam null cone. Note that this fixed radius cone would be used only at the first few steps of the observations. After that, at each step k, the position estimates of jammer $(\mathbf{x_k}, \mathbf{x_{k+1^-}})$ and the covariances $\mathbf{P_k}, \mathbf{P_{k+1^-}}$, are calculated. The optimal value of s_i is determined. The optimal s_i is then used to determine the beam null. Node i uses this beam null until the next observation at step $k+1$. At each step, the kalman filter algorithm is run which takes negligible amount of processing time as it deals with only matrix multiplications. The maximization of s_i also runs in the order of $\mathbb{N}$ as it checks with the neighbor of j whether j falls in the null region or not. The creation of the null depends on the hardware efficiency which takes time in the order of micro seconds [168, 169].

Algorithm 9: Algorithm for beam null at step k

```
measure angular position of jammer z_θ(k), z_φ(k)
if not enough observation then
    create beam null centering z_θ(k), z_φ(k) with a cone radius of r_th
else
    Calculate x_k, P_k, x_{k+1^-}, P_{k+1^-}
    Determine optimal s_i using eq. 6.53
    Create the beam null using optimal value of s_i
```

6.3.1.6 Defense against multiple jammers

So far we have discussed the calculation of a beam null for a single moving jammer. To defend multiple jammers, a node can adapt its gain pattern to include multiple nulls [160, 161]. However, determining the optimized beamnulls to defend multiple jammers is not very simple. It is not just creating multiple independent nulls for



(A) 2D representation in (θ, ϕ) plane

(B) 3D representation in (x, y, z) plane

FIGURE 6.22: Creating beam null for multiple jammer

each jammer. The nodes need to consider joint probability of link failure for link shadowing and probability of being attacked.

In this framework, a node maintain seperate kalman state vectors $(\mathbf{x}, \mathbf{F}, \mathbf{w}, \mathbf{z}, \mathbf{H})$ for each jammer. The node monitors the DoA jammer $z_\theta^v(k), z_\phi^v(k)$ at an interval of τ. Now, the node uses different standard deviation s^v to create the beam null border described in the earlier section. Figure 6.22 illustrates the scenario. The lower value of s^v reduces the beam null region that in effect reduces number of deactivated links. But it comes with a cost that there is a higher probability of node being jammed that, in effect, deactivates all links. Again, a very high value of s^v increases the number of deactivated links.

For multiple beamnull, with known beam nulls, a node i can assess the probability of the failure of link with j. Considering that j is not jammed, we can determine the probability that link ij fails can be obtained by modifying eq. 6.48 as:

$$\mathbb{P}(\text{ij fails}) = \begin{cases} 1 & if \quad \exists_{v \in V} \quad \varrho_j^v = True \\ \mathbb{P}(\text{i is jammed}) & otherwise \end{cases} \tag{6.47}$$

$$= (1 - \mathbb{P}(\text{node i is jammed})) \cup \varrho_j^v + \mathbb{P}(\text{node i is jammed}) \tag{6.48}$$

where ϱ_j^v checks if the link ij falls in the beamnull created for the jammer v.

The probability of a node successfully avoiding jamming is same as the probability that the jammers stay within the null during next transmission interval. As the error in the DoA estimation model is a normal distribution, we can say that probability of successful estimation would closely follow Chebyshev's inequality. In that case, if node i uses s_i^v standard deviation in eq. 6.25 and eq. 6.26 for calculating circle diameters for jammer v,

$$\mathbb{P}(\text{jammer } v \text{ satys in beam null}) \approx 1 - \frac{1}{(s_i^v)^2} \tag{6.49}$$

$$\mathbb{P}(\text{all the jammers stay in beamnulls}) \approx \prod_{v \in V}(1 - \frac{1}{(s_i^v)^2}) \tag{6.50}$$

For the optimization purpose, we can consider,

$$\mathbb{P}(\text{node i being jammed}) = 1 - \prod_{v \in V}(1 - \frac{1}{(s_i^v)^2}) \tag{6.51}$$

Thus, we can write,

$$\mathbb{P}(\text{ij fails}) = \cup\varrho_j^v + (1 - \cup\varrho_j^v)(1 - \prod_{v \in V}(1 - \frac{1}{(s_i^v)^2}))) \tag{6.52}$$

Considering $w_{ij};\ i, j \in \mathbb{N}$ denotes the weight for a link between i and j, If all links are equally important for a network then $w_{ij} = 1; \forall i, j$. Thus, the optimization problem becomes:

$$maximize \quad \sum_{j \in \mathbb{N}} w_{i,j} \left(1 - \cup\varrho_j^v - (1 - \cup\varrho_j^v)(1 - \prod_{v \in V}(1 - \frac{1}{(s_i^v)^2}))) \right) \tag{6.53}$$

The maximization problem stated above is a convex optimization problem that computes optimal s_i^v for each of the jammer at every step. The basic idea is it is sometimes better to choose lower confidence beam null for a jammer if that intended beamnull result in too many link failure.

6.3.2 Results

The proposed mechanism is evaluated in two methods. First, a custom built simulator is used to analyze the performance in terms of network connectivity, and then to observe the performance with upper layer protocol in more detail, we simulated the network in ns3.

6.3.2.1 Simulation setup

We built a custom simulator using python scripting language. It keeps track of the performance at every tick interval of value τ seconds. The parameters of the simulation are listed in Table 6.6. All nodes use Algorithm 9 individually to create the desired beam null. The nodes run the algorithm at every tick interval after measuring the local DoA of the jammer. The goal of this work is to evaluate the efficiency of the beam nulling mechanism. Hence, we assume the nodes are capable of detecting and measuring the DoA of the jammer through the mechanisms proposed in [73, 132–136]. Next, after running the algorithm, each node enters the communication phase, in which nodes outside the beam nulled region can communicate. In the meantime, if the attacker moves outside the nulled region of a node, then the node is considered jammed, preventing it to communicate with any previous neighbor. In the simulation, the nodes are positioned following a uniform random distribution. Same positions are used to compare different mobility models. The received power is calculated using the free space path loss model. A link between nodes is active only if both nodes are inside each other's communication

TABLE 6.6: Simulation parameters

Parameters	Symbol	Values
Simulation area		$10,000 \times 10,000 \times 4,000\ m^3$
Transmission power		30 dBm
Received Power cutoff		-78 dBm
Communication Frequency		2.4 GHz
Communication Radious		3146 m
Sensing interval	τ	50 ms
Simulation Time		500 s
Jammer'ś mobility model		Gauss-Markov
Transition covariance	$\mathbf{Q}$	4×4 identity matrix
Observation covariance	$\mathbf{R}$	2×2 identity matrix
Estimated initial state	$\widehat{\mathbf{x_0}}$	4×1 zero matrix
Initial state covariance	$\widehat{\mathbf{P_0}}$	4×4 identity matrix

range, and only if none of them are jammed. If nodes fall inside their neighbors null region, then the link is also considered broken. We use three different 3D mobility models for the jammer [170]: Random Walk, Random Direction, and Gauss-Markov; and compare the performance in each case.

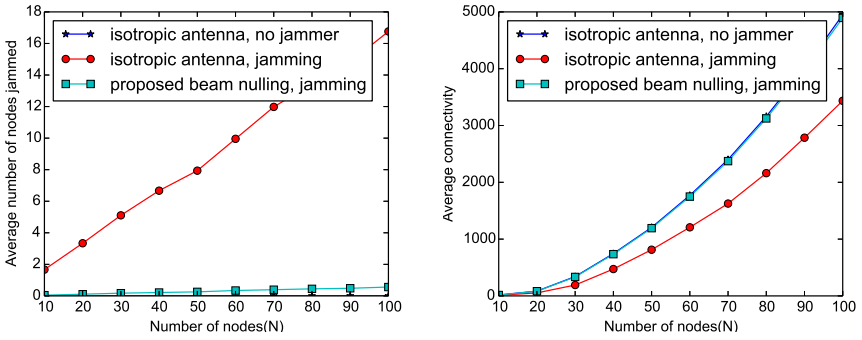
6.3.2.2 Performance metrics

We use four performance parameters:

i) *Average number of nodes jammed* defines the average number of nodes that are jammed during a simulation.

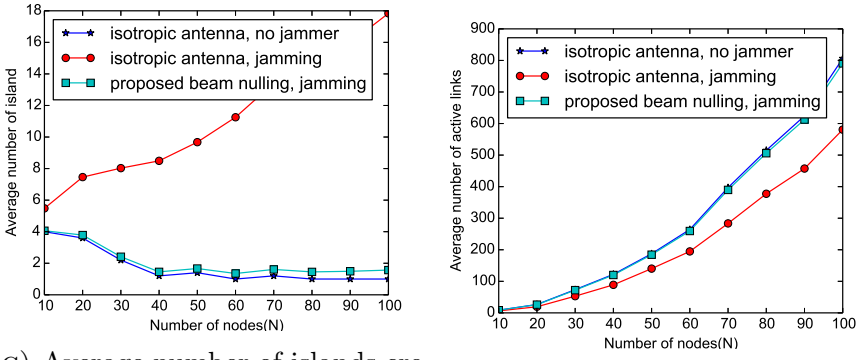
ii) We define *Connectivity* as the total number of connected pairs. This is a measure of how well connected the network is. It is defined as the summation of connected nodes. More precisely, connectivity of a network is $\frac{1}{2} \times (\sum_{i \in \mathbb{N}} \sum_{j \in \mathbb{N}} connected(i,j))$, where $connected(i,j) = 1$ if there exists at least one path from i to j, 0 otherwise.

iii) The third parameter is the *average number of active links*. A link between two nodes is considered to be deactivated if either of the corresponding nodes



(A) Average number of nodes attacked

(B) Average number of connectivity



(C) Average number of islands created

(D) Average number of active links

FIGURE 6.23: Simulation results

is attacked or one of the nodes fall in the beam null of the other. The total deactivated links are then divided by simulation time to obtain the average.

iv) The next performance parameter considered is the *average number of islands*. Sometimes a node or a group of nodes may be isolated from the rest of the network. The simulator counts the number of island present in the network at each tick. If a network is completely connected, the number of island is 1. The more islands, the more disrupted the network is.

6.3.2.3 Simulation varying number of nodes

To correctly evaluate the performance of the proposed scheme, two benchmark scenarios are considered. The first being *isotropic antenna without jammer*, where all nodes use isotropic antennas for communication at the absence of any jammer.

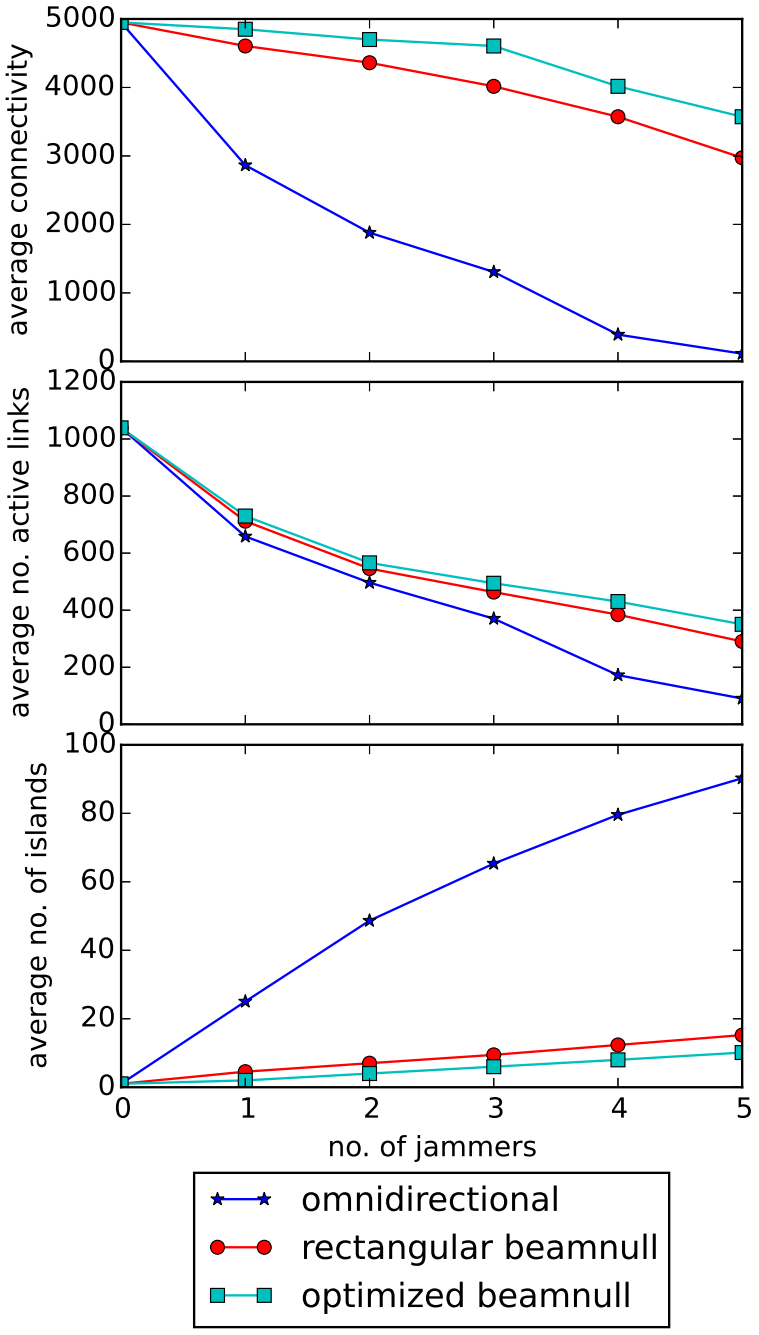


FIGURE 6.24: Results with multiple jammers

The second scenario is *isotropic antenna with a jamming* antenna where isotropic antennas are used for communication in the presence of a jammer. The third scenario uses the proposed adaptive beam nulling for avoiding the jammer.

Figure 6.23a depicts the comparison of average number of attacked nodes. We obtain the result for various node density. As the simulation area is fixed, the number

of nodes represented in x-axis reveals the node density. Nodes with isotropic antenna are vulnerable to jamming. As the density of nodes increases, more nodes are attacked as can be seen in the figure. The proposed mechanism uses adaptive nulling and avoid jamming. Some nodes will observe jamming due to inaccurate prediction. Note that a node would also experience jamming if the jammer was not in the vicinity in the previous step and, as a result, the node did not use beam nulling in that step. However, with the proposed scheme, nodes manage to keep the number of attacked nodes close to the ideal case of the no jammer scenario. The results show that with the proposed mechanism, a network can decrease the average number of jammed nodes up to 96.65%.

Figure 6.23b provides the performance of three scenarios in terms of average connectivity. At each tick, the simulator calculates the connectivity of the network. For a fully connected network with n nodes, the connectivity should indicate $\frac{n(n-1)}{2}$. For, a fully connected network with 100 nodes, the value should be 4950, which can be seen in the plot. It can be observed that with our proposed scheme, the network remains almost unaffected in terms of connectivity, as the connectivity is close to the benchmark case of no jammer. The plot reveals that with the proposed scheme, a network can increase its connectivity to 42.47% in presence of a jammer.

In Figure 6.23c, average number of islands are represented. For a very sparse network, where number of nodes in the network is very small, the network is not well connected. In these cases the network is not fully connected, and the network is divided into more than one islands. Even for the benchmark case of no jammer, there can exist more than one island. Multiple simulations with same number of nodes (N) are run with different random node positions are run, and the average is taken to obtain reliable results. For some generated graphs, the random position will make the network partitioned into islands. Thus, the average number of islands is not 1 even for high value of N. However, it can be clearly observed that

with our proposed algorithm, the network can keep the number of islands very close to the scenario of no jammer. The simulation reveals that with proposed beam nulling method, the number of islands can be decreased by 91.21%.

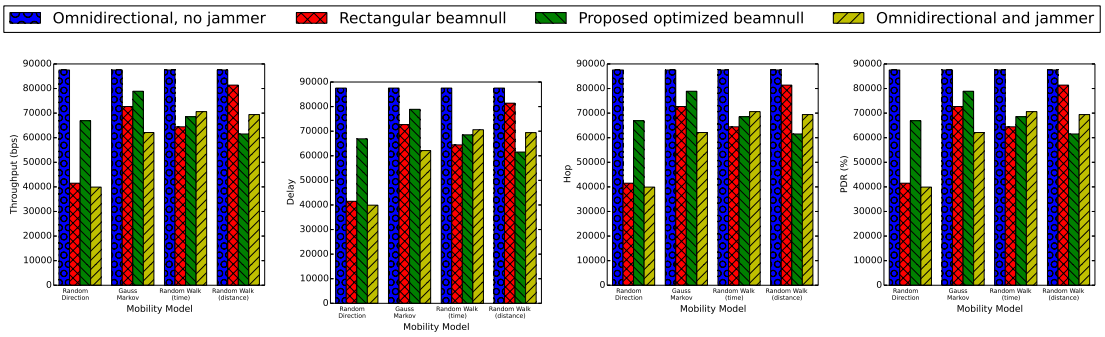
Figure 6.23d depicts the average number of active links in the network during simulation. The plot reveals that with proposed adaptive beamnulling, the network can retain more active links in the presence of a jammer. With the proposed mechanism, a network can retain 36.14% of its links that are jammed. It is noteworthy to mention that although there are many links deactivated, mostly due to neighbors being shadowed by beam null, the network remains connected as discussed in earlier. This proves that the proposed scheme successfully maintains the communication in jammed region.

6.3.2.4 Simulation with multiple jammers

We simulated the network with multiple jammers to illustrate the behavior of the adaptive beam nulling method as described in Section 6.3.1.6. The simulated network consists of 100 nodes, the rest of the parameters are kept same as before as listed in Table 6.6. In Figure 6.24 we plot the simulation results. Simulations are performed for different number of jammers in between 0 and 5, where 0 represents the case of no jammer as a benchmark of best case scenario.

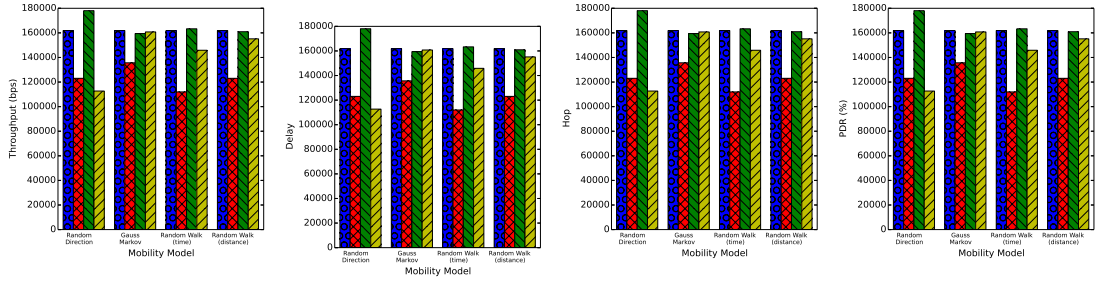
In our earlier work, we have proposed a framework that creates beamnulls whose borders are defined by lower and higher cutoffs in θ and ϕ direction. We call this framework as rectangular beamnull creation [171, 172].

Results show that using the optimized beam nulling improves the network connectivity when compared with the same case with omnidirectional antenna. In all the observed results, we see improvement from our earlier framework that creates rectangular beamnull.



(A) average throughput (B) average delay (C) average hop counts (D) packet delivery ratio

FIGURE 6.25: Simulation results with AODV as routing protocol



(A) average throughput (B) average delay (C) average hop counts (D) packet delivery ratio

FIGURE 6.26: Simulation results with DSDV as routing protocol

6.3.2.5 Simulation with upper layer protocols

To observe the effect the proposed mechanism may cause on the network as a whole, we simulate the new approach in a full stack simulator, namely ns-3. The simulation parameters are kept same as Section 6.2.2.5.

As mentioned before, we compare this beamnulling framework with schema proposed in Section 6.2, where the proposed approach estimates the speed and direction of the jammer based on the previous measurements. In that work, the beam null has a rectangular border in the (θ, ϕ) plane. We compare both beam nulling models with two benchmark scenarios. One is considered the best case scenario, where nodes use an omnidirectional antenna, and there is no jammer to disrupt the system. The second is the worst case, where nodes use omniderectional antennas, but there is a jammer, meaning nodes cannot adapt to avoid the jamming signal.

The results for AODV and DSDV routing protocols are shown in Figures 6.25 and 6.26 respectively. Both beamnullinf mechanisms improve the network connectivity regardless of the mobility model used for the jammer. The average throughput plots clearly show the improvement the new framework compared to our earlier work. Due to fluctuation of links in DSDV, a data packet may arrive more than one time to the destination, causing in some cases the throughput to be higher than when there is no attacker.

In both AODV and DSDV, the proposed mechanism provides delay close to the best case scenario. The worst case scenario of omnidirectional antenna in the presence of a jammer results in several link failures, and consequently some sources or destination nodes are prevented from communicating. Due to the less amount of traffic (from the blocked nodes), the load and congestion on the rest of the nodes are reduced. Thus, the computed delay is also reduced. However, the proposed mechanism allows nodes close to the jamming radius to stay active, but with the loss of some links some rerouting is necessary to maintain the connectivity. As a result the average hop count and delay increase. However, the results clearly indicate the benefits of having optimal beamnull over the rectangular beamnull.

6.4 Summary

This chapter analyzes the usability of adaptive beamnulling as a jamming mitigation technique in 3D mesh networks. It proposes two frameworks for adaptive beam nulling in multihop ad hoc networks as a mitigation technique against a moving jammer. Performance of the proposed framework is studied through physical layer and full-stack network simulations of various network topologies and mobility models of the jammer in both 2-dimensional and 3-dimensional environments. Obtained results indicate that employing this framework leads to significant improvements in survivability of the links and connectivity over the performance of

networks with omnidirectional interfaces. Also, to increase the accuracy of the simulated models for practical implementations, effects of varying inherent errors on the performance of a beam nulling ad hoc network is studied.

Chapter 7

Conclusions

In this thesis, we have focused on how next generation wireless networks can defend against intelligent adversaries. As a first step, we focused on cognitive radio networks. In the second step, we investigated the applicability of adaptive beamnulling in distributed 3D mesh networks to avoid jamming from jammer that can move in all three dimensions.

In the first phase, we developed a comprehensive testbed for dynamic spectrum access network using off-the-shelf software-defined radios. We have proposed techniques to obtain duplex communication using single USRP devices. The duplex communication is achieved by separating the channels in the frequency domain and applying proper filtering. We have developed a scheme where nodes in a mesh network can acquire spectrum or change their bandwidth online. Since the online spectrum allocation by multiple nodes results in spectrum wastage, online defragmentation is proposed as a method of increasing spectrum utilization in channel-aggregating DSA radio networks. The efficiency of this method was investigated in three different network scenarios; Infrastructure, distributed, and semi-centralized. By including parameters retrieved from a proof-of-concept prototype into simulations, realistic comparisons of the three scenarios with regards

to the effectiveness of the presented algorithm is presented. It was concluded that regardless of the situation, defragmentation provides better performance regarding spectral efficiency and throughput.

In the second phase, we propose CR-Honeynet, a CRN sustenance mechanism, which exploits the fact that an intelligent and rational attacker aims for certain transmission characteristics to gain the highest impact by jamming. The stochastic learning model presented shows that the honeynet can confidently learn an attacker's strategy and dynamically evolve with the attacker's strategy changes. The mechanism efficiently lures the attacker towards the active decoy trap, and thus bypassing attacks on legitimate SU communications. We simulated the performance of a CRN based on queuing model with fixed vacation. The model deals with the periodic sensing of a cognitive cycle as a fixed periodic vacation. We show that CR-honeynet is useful for preventing the jamming attack; however assigning honeynode without considering the queuing delay associated with it causes performance degradation. Under such circumstances, we have shown that dynamic assignment of honeynode is crucial from the system's performance perspective. We propose state dependent honeynode selection strategies at the beginning of every transmission cycle where the honeynode selection can be made by choosing the SU that has the highest probability of emptying the queue. We have proposed an optimization criterion that minimizes overall queuing delay while maintaining good fairness among competing nodes. The effectiveness of CR-Honeynet is proved using a state-of-the-art testbed.

In the third phase, we focused on another vulnerability in CRN with heterogeneous channels, known as induction attack. An attacker can target channels intelligently to push the defender to use non-optimal channels. We used a game to model the actions of choosing channels for transmission (by the user) and attack (by the attacker). We describe a closed form solution for the game when the channel

utilities are known and fixed. We show that there exist a saddle point for the learning period of both the defender and the attacker.

In the final stage, we considered the case where the jammer is capable of autonomous movements in all three directions. A jammer with 3D movement can cause more impact on a network by jamming selective nodes. We proposed distributed adaptive beamnulling frameworks to defend jammers. The nodes independently scan for jamming periodically and predict the most likely path of the jammer within the next sensing interval. We propose optimal beam null border determination criteria that increases the probability to keep the jammer in null while minimizing the link failure due to shadowing. The simulation carried out in both the custom simulator and ns3 proves the effectiveness of the scheme.

Bibliography

[1] "A new spectrum observatory in brussels: providing data to inform policy decisions." http://www.microsoft.eu/2013/01/28/a-new-spectrum-observatory-in-brussels-providing-data-to-inform-policy-decisions.

[2] R. G. Brown and P. Y. Hwang, *Introduction to Random Signals and Applied Kalman Filtering with Matlab Exercises, 4th Edition*. Wiley, 2012.

[3] S. Sengupta, M. Chatterjee, and S. Ganguly, "Improving quality of voip streams over wimax," *Computers, IEEE Transactions on*, vol. 57, no. 2, pp. 145–156, 2008.

[4] R. G. Cole and J. H. Rosenbluth, "Voice over IP performance monitoring," *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 2, pp. 9–24, 2001.

[5] L. Ding and R. A. Goubran, "Speech quality prediction in voip using the extended e-model," in *Global Telecommunications Conference, 2003. GLOBECOM'03. IEEE*, vol. 7, pp. 3974–3978, IEEE, 2003.

[6] J. Q. Walker, "Assessing voip call quality using the e-model."

[7] V. Balan and L. Eggert, "An experimental evaluation of voice-over-ip quality over the datagram congestion control protocol," *School of Engineering and Scienze International University Bremen*, 2006.

[8] "Voice Over IP - per call bandwidth consumption."

[9] I. F. Akyildiz, B. F. Lo, and R. Balakrishnan, "Cooperative spectrum sensing in cognitive radio networks: A survey," *Physical Communication*, vol. 4, no. 1, pp. 40 – 62, 2011.

[10] A. Fragkiadakis, E. Tragos, and I. Askoxylakis, "A survey on security threats and detection techniques in cognitive radio networks," *IEEE Communications Surveys Tutorials*, vol. 15, no. 1, pp. 428–445, 2013.

[11] A. Raschella, *COGNITIVE MANAGEMENT FRAMEWORKs AND SPECTRUM MANAGEMENT STRATEGIES EXPLOITING COGNITIVE RADIO PARADIGM*. PhD thesis, Universitat Politecnica de Catalunya, May 2015.

[12] "IEEE draft standard for information technology -telecommunications and information exchange between systems - wireless regional area networks (WRAN) - specific requirements - part 22: Cognitive wireless ran MAC & PHY specifications: Policies and procedures for operation in the TV bands." IEEE P802.22/D2.0, 2011.

[13] D. K. Tosh, S. K. Udgata, and S. L. Sabat, "Cognitive radio parameter adaptation using multi-objective evolutionary algorithm," in *2011 International Conference on Soft Computing for Problem Solving (SocProS 2011)*, vol. 1, pp. 737–748, Springer India, 2012.

[14] S. Bhattacharjee, S. Sengupta, and M. Chatterjee, "Vulnerabilities in cognitive radio networks: A survey," *Computer Communications*, vol. 36, no. 13, pp. 1387–1398, 2013.

[15] T. X. Brown and A. Sethi, "Potential cognitive radio denial-of-service vulnerabilities and protection countermeasures: A multi-dimensional analysis and assessment," *Mobile Networks and Applications*, vol. 13, no. 5, pp. 516–532, 2008.

[16] L. Jiao, V. Pla, and F. Y. Li, "Analysis on channel bonding/aggregation for multi-channel cognitive radio networks," in *Wireless Conference (EW), 2010 European*, pp. 468–474, IEEE, 2010.

[17] D. Chen, Q. Zhang, and W. Jia, "Aggregation aware spectrum assignment in cognitive ad-hoc networks," in *Cognitive Radio Oriented Wireless Networks and Communications, 2008. CrownCom 2008. 3rd International Conference on*, pp. 1–6, IEEE, 2008.

[18] F. Huang, W. Wang, H. Luo, G. Yu, and Z. Zhang, "Prediction-based spectrum aggregation with hardware limitation in cognitive radio networks," in *Vehicular Technology Conference (VTC 2010-Spring), 2010 IEEE 71st*, pp. 1–5, IEEE, 2010.

[19] L. Yang, W. Hou, L. Cao, B. Y. Zhao, and H. Zheng, "Supporting demanding wireless applications with frequency-agile radios.," in *Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation*, NSDI 2010, pp. 65–80, 2010.

[20] S. Anand, S. Sengupta, K. Hong, K. Subbalakshmi, R. Chandramouli, and H. Cam, "Exploiting channel fragmentation and aggregation/ bonding to create security vulnerabilities," *IEEE Transactions on Vehicular Technology*, 2014.

[21] K. Pelechrinis, M. Iliofotou, and S. V. Krishnamurthy, "Denial of service attacks in wireless networks: The case of jammers," *Communications Surveys & Tutorials, IEEE*, vol. 13, no. 2, pp. 245–257, 2011.

[22] Q. Wang, K. Ren, and P. Ning, "Anti-jamming communication in cognitive radio networks with unknown channel statistics," in *Network Protocols (ICNP), 2011 19th IEEE International Conference on*, pp. 393–402, IEEE, 2011.

[23] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures," *Ad hoc networks*, vol. 1, no. 2, pp. 293–315, 2003.

[24] K. Levy and F. J. Vázquez-Abad, "Change-point monitoring for online stochastic approximations," *Automatica*, vol. 46, no. 10, pp. 1657 – 1674, 2010.

[25] L. De Filippis, G. Guglieri, and F. Quagliotti, "Path planning strategies for UAVs in 3D environments," *Journal of Intelligent & Robotic Systems*, vol. 65, no. 1-4, pp. 247–264, 2012.

[26] M. Khan, S. Bhunia, M. Yuksel, and S. Sengupta, "Los discovery in 3d for highly directional transceivers," in *Military Communications Conference (MILCOM)*, 2016.

[27] S. Bhunia, M. Khan, S. Sengupta, and M. Yuksel, "Los discovery for highly directional full duplex rf/fso transceivers," in *Military Communications Conference (MILCOM)*, 2016.

[28] P. A. Regis, S. Bhunia, and S. Sengupta, "Implementation of 3d obstacle compliant mobility models for uav networks in ns-3," in *The Workshop on ns-3 (WNS3), Seattle, Washington*, 2016.

[29] A. Farago, "Graph theoretic analysis of ad-hoc network vulnerability," in *WiOpt'03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, 2003.

[30] P. A. Regis, S. Bhunia, and S. Sengupta, "Enhancing performance and longevity of multi-radio multi-channel hetnets through dynamic path-assignment," in *2017 International Conference on Computing, Networking and Communications (ICNC)*, IEEE, 2017.

[31] A. Yahya, O. Sidek, J. Mohamad-Saleh, and C. M. D. E. Center, "Performance analyses of fast frequency hopping spread spectrum and jamming systems.," *Int. Arab J. Inf. Technol.*, vol. 5, no. 2, pp. 115–119, 2008.

[32] L. B. Milstein, "Interference rejection techniques in spread spectrum communications," *Proceedings of the IEEE*, vol. 76, no. 6, pp. 657–671, 1988.

[33] B. D. Van Veen and K. M. Buckley, "Beamforming: A versatile approach to spatial filtering," *IEEE assp magazine*, vol. 5, no. 2, pp. 4–24, 1988.

[34] K.-B. Yu, "Adaptive beamforming for satellite communication with selective earth coverage and jammer nulling capability," *Signal Processing, IEEE Transactions on*, vol. 44, no. 12, pp. 3162–3166, 1996.

[35] W. C. Cummings, "An adaptive nulling antenna for military," *Lincoln Laboratory Journal,1992,*, vol. 5, no. 2, 1992.

[36] G. Okamoto, "Jammer nulling via low complexity blind beamforming algorithm," in *Antennas and Propagation Society International Symposium, 2007 IEEE*, pp. 25–28, IEEE, 2007.

[37] W. Zhu, B. Daneshrad, J. Bhatia, H.-S. Kim, D. Liu, K. Mohammed, R. Prabhu, S. Sasi, and A. Shah, "MIMO systems for military communications," in *Military Communications Conference, 2006. MILCOM 2006. IEEE*, pp. 1–7, IEEE, 2006.

[38] D. Mingjie, P. Xinjian, Y. Fang, and L. Jianghong, "Research on the technology of adaptive nulling antenna used in anti-jam GPS," in *Radar, 2001 CIE International Conference on, Proceedings*, pp. 1178–1181, IEEE, 2001.

[39] G. K. Rao and R. S. H. Rao, "Status study on sustainability of satellite communication systems under hostile jamming environment," in *India Conference (INDICON), 2011 Annual IEEE*, pp. 1–7, IEEE, 2011.

[40] Q. Zhao and B. M. Sadler, "A survey of dynamic spectrum access," *Signal Processing Magazine, IEEE*, vol. 24, no. 3, pp. 79–89, 2007.

[41] E. Chen and C. Chu, "Channel estimation for nc-ofdm systems based on subspace pursuit algorithm," in *Signal Processing (ICSP), 2012 IEEE 11th International Conference on*, vol. 1, pp. 88–91, IEEE, 2012.

[42] D. K. Tosh and S. Sengupta, "Heterogeneous access network (s) selection in multi-interface radio devices," in *Pervasive Computing and Communication Workshops (PerCom Workshops), 2015 IEEE International Conference on*, pp. 117–122, IEEE, 2015.

[43] D. K. Tosh and S. Sengupta, "Self-coexistence in cognitive radio networks using multi-stage perception learning," in *Vehicular Technology Conference (VTC Fall), 2013 IEEE 78th*, pp. 1–5, IEEE, 2013.

[44] J. Lin, L. Shen, N. Bao, B. Su, Z. Deng, and D. Wang, "Channel characteristic aware spectrum aggregation algorithm in cognitive radio networks," in *Local Computer Networks (LCN), 2011 IEEE 36th Conference on*, pp. 634–639, IEEE, 2011.

[45] "GNU Radio." http://gnuradio.org/redmine/projects/gnuradio/wiki.

[46] "Usrp kit." https://www.ettus.com/product/details/UN200-KIT.

[47] R. Chen, J.-M. Park, Y. T. Hou, and J. H. Reed, "Toward secure distributed spectrum sensing in cognitive radio networks," *Communications Magazine, IEEE*, vol. 46, no. 4, pp. 50–55, 2008.

[48] X. Zhang and C. Li, "The security in cognitive radio networks: a survey," in *Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly*, IWCMC '09, (New York, NY, USA), pp. 309–313, ACM, 2009.

[49] S. E. Frankel, B. Eydt, L. Owens, and K. A. Scarfone, "Sp 800-97. establishing wireless robust security networks: A guide to ieee 802.11i," tech. rep., National Institute of Standards & Technology, Gaithersburg, MD, United States, 2007.

[50] S. Sanyal, R. Bhadauria, and C. Ghosh, "Secure communication in cognitive radio networks," in *Computers and Devices for Communication, 2009. CODEC 2009. 4th International Conference on*, pp. 1–4, 2009.

[51] T. X. Brown and A. Sethi, "Potential cognitive radio denial-of-service vulnerabilities and protection countermeasures: a multi-dimensional analysis and assessment," *Mob. Netw. Appl.*, vol. 13, pp. 516–532, Oct. 2008.

[52] T. Clancy and N. Goergen, "Security in cognitive radio networks: Threats and mitigation," in *Cognitive Radio Oriented Wireless Networks and Communications, 2008. CrownCom 2008. 3rd International Conference on*, pp. 1–8, 2008.

[53] J. Burbank, "Security in cognitive radio networks: The required evolution in approaches to wireless network security," in *Cognitive Radio Oriented Wireless Networks and Communications, 2008. CrownCom 2008. 3rd International Conference on*, pp. 1–7, 2008.

[54] A. Sethi and T. Brown, "Hammer model threat assessment of cognitive radio denial of service attacks," in *New Frontiers in Dynamic Spectrum Access Networks, 2008. DySPAN 2008. 3rd IEEE Symposium on*, pp. 1–12, 2008.

[55] N. R. Prasad, "Secure cognitive networks," in *Wireless Technology, 2008. EuWiT 2008. European Conference on*, pp. 107–110, IEEE, 2008.

[56] C. N. Mathur and K. Subbalakshmi, "Digital signatures for centralized dsa networks," in *Consumer Communications and Networking Conference, 2007. CCNC 2007. 4th IEEE*, pp. 1037–1041, IEEE, 2007.

[57] R. Chen, J.-M. Park, and J. H. Reed, "Defense against primary user emulation attacks in cognitive radio networks," *Selected Areas in Communications, IEEE Journal on*, vol. 26, no. 1, pp. 25–37, 2008.

[58] F. R. Yu, H. Tang, M. Huang, Z. Li, and P. C. Mason, "Defense against spectrum sensing data falsification attacks in mobile ad hoc networks with cognitive radios," in *Military Communications Conference, 2009. MILCOM 2009. IEEE*, pp. 1–7, IEEE, 2009.

[59] K. Bian and J.-M. J. Park, "Security vulnerabilities in ieee 802.22," in *Proceedings of the 4th Annual International Conference on Wireless Internet*, WICON '08, (ICST, Brussels, Belgium, Belgium), pp. 9:1–9:9, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008.

[60] O. León, J. Hernández-Serrano, and M. Soriano, "Securing cognitive radio networks," *International Journal of Communication Systems*, vol. 23, no. 5, pp. 633–652, 2010.

[61] A. Fragkiadakis, E. Tragos, and I. Askoxylakis, "A survey on security threats and detection techniques in cognitive radio networks," *Communications Surveys Tutorials, IEEE*, vol. 15, no. 1, pp. 428–445, 2013.

[62] Y. Xing, C. N. Mathur, M. Haleem, R. Chandramouli, and K. Subbalakshmi, "Priority based dynamic spectrum access with qos and interference temperature constraints," in *Communications, 2006. ICC'06. IEEE International Conference on*, vol. 10, pp. 4420–4425, IEEE, 2006.

[63] A. Naveed and S. S. Kanhere, "Nis07-5: Security vulnerabilities in channel assignment of multi-radio multi-channel wireless mesh networks," in *Global Telecommunications Conference, 2006. GLOBECOM'06. IEEE*, pp. 1–5, IEEE, 2006.

[64] A. S. Rawat, P. Anand, H. Chen, and P. K. Varshney, "Collaborative spectrum sensing in the presence of byzantine attacks in cognitive radio networks," *Signal Processing, IEEE Transactions on*, vol. 59, no. 2, pp. 774–786, 2011.

[65] J. Newsome, E. Shi, D. Song, and A. Perrig, "The sybil attack in sensor networks: analysis & defenses," in *Proceedings of the 3rd international symposium on Information processing in sensor networks*, pp. 259–268, ACM, 2004.

[66] D. K. Tosh, S. Sengupta, S. Mukhopadhyay, C. Kamhoua, and K. Kwiat, "Game theoretic modeling to enforce security information sharing among firms," in *IEEE 2nd International Conference on Cyber Security and Cloud Computing (CSCloud)*, pp. 7–12, 2015.

[67] D. K. Tosh, M. Molloy, S. Sengupta, C. A. Kamhoua, and K. A. Kwiat, "Cyber-investment and cyber-information exchange decision modeling," in *IEEE 7th Intl. Symposium on Cyberspace Safety and Security*, pp. 1219–1224, 2015.

[68] C. Kamhoua, A. Martin, D. K. Tosh, K. Kwiat, C. Heitzenrater, and S. Sengupta, "Cyber-threats information sharing in cloud computing: A game theoretic approach," in *IEEE 2nd International Conference on Cyber Security and Cloud Computing (CSCloud)*, pp. 382–389, 2015.

[69] S. Misra, S. K. Dhurandher, A. Rayankula, and D. Agrawal, "Using honeynodes for defense against jamming attacks in wireless infrastructure-based networks," *Computers & electrical engineering*, vol. 36, no. 2, pp. 367–382, 2010.

[70] 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*, pp. 80–89, ACM, 2004.

[71] H. Kim and K. G. Shin, "In-band spectrum sensing in cognitive radio networks: energy detection or feature detection?," in *Proceedings of the 14th ACM international conference on Mobile computing and networking*, pp. 14–25, ACM, 2008.

[72] V. Chatzigiannakis, G. Androulidakis, K. Pelechrinis, S. Papavassiliou, and V. Maglaris, "Data fusion algorithms for network anomaly detection: classification and evaluation," in *Networking and Services, 2007. ICNS. Third International Conference on*, pp. 50–50, IEEE, 2007.

[73] C. Sorrells, L. Qian, and H. Li, "Quickest detection of denial-of-service attacks in cognitive wireless networks," in *Homeland Security (HST), 2012 IEEE Conference on Technologies for*, pp. 580–584, IEEE, 2012.

[74] S. Bhunia and S. Sengupta, "Feasibility of channel hopping in jamming attack," *IEEE TCSIM Newsletter*, no. 19, pp. 2–5, 2013.

[75] V. Navda, A. Bohra, S. Ganguly, and D. Rubenstein, "Using channel hopping to increase 802.11 resilience to jamming attacks," in *26th IEEE International Conference on Computer Communications (INFOCOM)*, pp. 2526–2530, IEEE, 2007.

[76] W. Xu, K. Ma, W. Trappe, and Y. Zhang, "Jamming sensor networks: attack and defense strategies," *Network, IEEE*, vol. 20, no. 3, pp. 41–47, 2006.

[77] C. Sorrells, P. Potier, L. Qian, and X. Li, "Anomalous spectrum usage attack detection in cognitive radio wireless networks," in *IEEE International Conference on Technologies for Homeland Security (HST), 2011*, pp. 384–389, IEEE, 2011.

[78] Z. Jin, S. Anand, and K. Subbalakshmi, "Detecting primary user emulation attacks in dynamic spectrum access networks," in *Communications, 2009. ICC'09. IEEE International Conference on*, pp. 1–5, IEEE, 2009.

[79] C. Pöpper, M. Strasser, and S. Čapkun, "Anti-jamming broadcast communication using uncoordinated spread spectrum techniques," *Selected Areas in Communications, IEEE Journal on*, vol. 28, no. 5, pp. 703–715, 2010.

[80] H. Rahul, F. Edalat, D. Katabi, and C. G. Sodini, "Frequency-aware rate adaptation and mac protocols," in *15th annual international conference on Mobile computing and networking (MobiCom)*, 2009.

[81] H. Lee, S. Vahid, and K. Moessner, "The study on spectrum/channel fragmentation from dynamic spectrum aggregation in crns," in *European Conference on Networks and Communications (EuCNC)*, pp. 1–5, 2014.

[82] S. Martello and P. Toth, *Knapsack problems*. Wiley New York, 1990.

[83] K. R. Chowdhury and I. F. Akyldiz, "Ofdm-based common control channel design for cognitive radio ad hoc networks," *Mobile Computing, IEEE Transactions on*, vol. 10, no. 2, pp. 228–238, 2011.

[84] C. Doerr, D. Grunwald, and D. C. Sicker, "Dynamic control channel management in presence of spectrum heterogeneity," in *Military Communications Conference, 2008. MILCOM 2008. IEEE*, IEEE.

[85] M.-R. Kim and S.-J. Yoo, "Distributed coordination protocol for common control channel selection in multichannel ad-hoc cognitive radio networks," in *Wireless and Mobile Computing, Networking and Communications, 2009. WIMOB 2009. IEEE International Conference on*, pp. 227–232, IEEE, 2009.

[86] A. A. Abbasi and M. Younis, "A survey on clustering algorithms for wireless sensor networks," *Computer communications*, vol. 30, no. 14, pp. 2826–2841, 2007.

[87] S. Bhunia, V. Behzadan, and S. Sengupta, "Enhancement of spectrum utilization in non-contiguous dsa with online defragmentation," in *Military Communications Conference, MILCOM 2015-2015 IEEE*, pp. 432–437, IEEE, 2015.

[88] J. Burbank, "Security in cognitive radio networks: The required evolution in approaches to wireless network security," in 3^{rd} *International Conference on Cognitive Radio Oriented Wireless Networks and Communications.*, pp. 1 –7, may 2008.

[89] S. Misra, S. K. Dhurandher, A. Rayankula, and D. Agrawal, "Using honeynodes for defense against jamming attacks in wireless infrastructure-based networks," *Computers and Electrical Engineering*, vol. 36, no. 2, pp. 367 – 382, 2010.

[90] "Wi-spy spectrum analyzer." http://www.metageek.net/products/wi-spy/.

[91] D. Tosh, S. Sengupta, C. A. Kamhoua, and K. A. Kwiat, "Establishing evolutionary game models for cyber security information exchange (cybex)," *Elsevier Journal of Computer and System Sciences*, 2016.

[92] D. Tosh, S. Sengupta, C. Kamhoua, K. Kwiat, and A. Martin, "An evolutionary game-theoretic framework for cyber-threat information sharing," in *Communications (ICC), 2015 IEEE International Conference on*, pp. 7341–7346, IEEE, 2015.

[93] K. C. Madan, "An m/g/1 queue with optional deterministic server vacations," *Metron - International Journal of Statistics*, vol. 0, no. 3-4, pp. 83–95, 1999.

[94] S. M. Ross, *Introduction to Probability Models,* 10^{th} *Ed.* Academic Press, Dec. 2009.

[95] G. Choudhury and L. Tadj, "An M/G/1 queue with two phases of service subject to the server breakdown and delayed repair," *Applied Mathematical Modelling*, vol. 33, no. 6, pp. 2699–2709, 2009.

[96] S. Wang, J. Zhang, and L. Tong, "Delay analysis for cognitive radio networks with random access: a fluid queue view," in *INFOCOM, 2010 Proceedings IEEE*, pp. 1–9, IEEE, 2010.

[97] R. E. Bechhofer, S. Elmaghraby, and N. Morse, "A single-sample multiple-decision procedure for selecting the multinomial event which has the highest probability," *The Annals of Mathematical Statistics*, pp. 102–119, 1959.

[98] A. Gelfand, J. Glaz, L. Kuo, and T.-M. Lee, "Inference for the maximum cell probability under multinomial sampling," *Naval Research Logistics (NRL)*, vol. 39, no. 1, pp. 97–114, 1992.

[99] S. Ross, *Simulation*. Elsevier Science, 2012.

[100] L. Cai, X. Shen, J. Mark, L. Cai, and Y. Xiao, "Voice capacity analysis of wlan with unbalanced traffic," *IEEE Transactions on Vehicular Technology*, may 2006.

[101] L. Sun and E. Ifeachor, "Voice quality prediction models and their application in voip networks," *IEEE Transactions on Multimedia*, vol. 8, pp. 809 –820, aug. 2006.

[102] A. Mukhopadhyay, T. Chakraborty, S. Bhunia, I. S. Misra, and S. K. Sanyal, "Study of enhanced voip performance under congested wireless network scenarios," in *2011 Third International Conference on Communication Systems and Networks (COMSNETS 2011)*, pp. 1–7, IEEE, 2011.

[103] A. Kundu, I. S. Misra, S. K. Sanyal, and S. Bhunia, "Voip performance over broadband wireless networks under static and mobile environments,"

International Journal of Wireless & Mobile Networks, vol. 2, no. 4, pp. 82–93, 2010.

[104] A. Kundu, S. Bhunia, I. S. Misra, and S. K. Sanyal, "Comparison of voip performance over wimax, wlan and wimax-wlan integrated network using opnet," *Recent Trends in Networks and Communications*, pp. 316–325, 2010.

[105] D. O. Vic Grout, Stuart Cunningham and R. Hebblewhite, "A note on the distribution of packet arrivals in high-speed data networks," in *Proceedings of IADIS International Conference WWW/Internet*, 2004.

[106] R. Jain and S. A. Routhier, "Packet trains-measurements and a new model for computer network traffic," *IEEE Journal on Selected Areas in Communication*, vol. 6, pp. 986–995, September 1986.

[107] M. Wilson, "A historical view of network traffic models." http://www.cse.wustl.edu /~jain/cse567-06/ftp/traffic_models2. Accessed: 12/05/2012.

[108] D. R. Dennis Guster and R. Sundheim, "Evaluating computer network packet inter-arrival distributions," *Encyclopedia of Information Science & Technology*, 2009.

[109] T. Chakraborty, A. Mukhopadhyay, S. Bhunia, I. S. Misra, and S. K. Sanyal, "Analysis and enhancement of qos in cognitive radio network for efficient voip performance," in *Information and Communication Technologies (WICT), 2011 World Congress on*, pp. 904–909, IEEE, 2011.

[110] T. Chakraborty, A. Mukhopadhyay, S. Bhunia, I. S. Misra, and S. K. Sanyal, "Optimizing voip call in diverse network scenarios using state-space search technique," *Advances in Computing and Information Technology*, pp. 231–242, 2011.

[111] R. Kaas, M. Goovaerts, J. Dhaene, and M. Denuit, *Modern actuarial risk theory*, vol. 328. Springer, 2001.

[112] S. M. Ross, *Simulation, 5th Edision.* Academic Press, October 2012.

[113] S. Bhunia, I. S. Misra, S. K. Sanyal, and A. Kundu, "Performance study of mobile wimax network with changing scenarios under different modulation and coding," *International Journal of Communication Systems*, vol. 24, no. 8, pp. 1087–1104, 2011.

[114] S. Bhunia, A. Kundu, I. S. Misra, and S. K. Sanyal, "Reducing hand-off latency in wimax network using cross layer information," in *Advances in Computer Engineering (ACE), 2010 International Conference on*, pp. 346–348, IEEE, 2010.

[115] A. Mukhopadhyay, T. Chakraborty, S. Bhunia, I. S. Misra, and S. K. Sanyal, "An adaptive jitter buffer playout algorithm for enhanced voip performance," *Advances in Computing and Information Technology*, pp. 219–230, 2011.

[116] T. Chakraborty, A. Mukhopadhyay, S. Bhunia, I. S. Misra, and S. K. Sanyal, "An optimization technique for improved voip performance over wireless lan," *Journal of Networks*, vol. 7, no. 3, pp. 480–493, 2012.

[117] S. Bhunia, *Performance Evaluation of WiMAX Network in Aspect of Modulation and Coding Schemes and Hand-off using OPNET.* PhD thesis, JADAVPUR UNIVERSITY KOLKATA, 2010.

[118] R. Jain, D.-M. Chiu, and W. R. Hawe, *A quantitative measure of fairness and discrimination for resource allocation in shared computer system.* Eastern Research Laboratory, Digital Equipment Corporation, 1984.

[119] IEEE Computer Society, *IEEE Std 802.22a-2014: Cognitive Wireless RAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications:*

Policies and Procedures for Operation in the TV Bands, Amendment 1: Management and Control Plane Interfaces and Procedures and Enhancement to the Management Information Base (MIB), 2014.

[120] IEEE Computer Society, *IEEE Std 802.22b-2015: Cognitive Wireless RAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Policies and Procedures for Operation in the TV Bands, Amendment 2: Enhancement for Broadband Services and Monitoring Applications*, 2015.

[121] A. Fragkiadakis, E. Tragos, and I. Askoxylakis, "A survey on security threats and detection techniques in cognitive radio networks," *Communications Surveys Tutorials, IEEE*, vol. 15, pp. 428–445, First 2013.

[122] R. K. Sharma and D. B. Rawat, "Advances on security threats and countermeasures for cognitive radio networks: A survey," *Communications Surveys & Tutorials, IEEE*, vol. 17, no. 2, pp. 1023–1043, 2015.

[123] S. Bhattacharjee, S. Sengupta, and M. Chatterjee, "Vulnerabilities in cognitive radio networks: A survey," *Elsevier Computer Communications*, vol. 36, no. 13, pp. 1387–1398, 2013.

[124] M. T. Masonta, M. Mzyece, and N. Ntlatlapa, "Spectrum decision in cognitive radio networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1088–1107, 2013.

[125] M. Naeem, A. Anpalagan, M. Jaseemuddin, and D. C. Lee, "Resource allocation techniques in cooperative cognitive radio networks," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 2, pp. 729–744, 2014.

[126] V. K. Tumuluru, P. Wang, D. Niyato, and W. Song, "Performance analysis of cognitive radio spectrum access with prioritized traffic," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 4, pp. 1895–1906, 2012.

[127] Y. Wu, B. Wang, K. R. Liu, and T. C. Clancy, "Anti-jamming games in multi-channel cognitive radio networks," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 1, pp. 4–15, 2012.

[128] Z. Shu, Y. Qian, and S. Ci, "On physical layer security for cognitive radio networks," *IEEE Network*, vol. 27, no. 3, pp. 28–33, 2013.

[129] C. Chen, M. Song, C. Xin, and J. Backens, "A game-theoretical anti-jamming scheme for cognitive radio networks," *IEEE Network*, vol. 27, no. 3, pp. 22–27, 2013.

[130] S. Bhunia, S. Sengupta, and F. Vazquez-Abad, "Cr-honeynet: A learning & decoy based sustenance mechanism against jamming attack in crn," in *Military Communications Conference (MILCOM), 2014 IEEE*, pp. 1173–1180, IEEE, 2014.

[131] S. Bhunia, X. Su, S. Sengupta, and F. Vázquez-Abad, "Stochastic model for cognitive radio networks under jamming attacks and honeypot-based prevention," in *International Conference on Distributed Computing and Networks (ICDCN)*, pp. 438–452, Springer Berlin Heidelberg, 2014.

[132] J. T. Chiang and Y.-C. Hu, "Cross-layer jamming detection and mitigation in wireless broadcast networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 19, no. 1, pp. 286–298, 2011.

[133] M. Spuhler, D. Giustiniano, V. Lenders, M. Wilhelm, and J. B. Schmitt, "Detection of reactive jamming in DSSS-based wireless communications," *Wireless Communications, IEEE Transactions on*, vol. 13, no. 3, pp. 1593–1603, 2014.

[134] H. L. Van Trees, *Detection, estimation, and modulation theory*. John Wiley & Sons, 2004.

[135] M. Jin, G. Liao, and J. Li, "Joint dod and doa estimation for bistatic mimo radar," *Signal Processing*, vol. 89, no. 2, pp. 244–251, 2009.

[136] X. Zhang, L. Xu, L. Xu, and D. Xu, "Direction of departure (dod) and direction of arrival (doa) estimation in mimo radar with reduced-dimension music," *Communications Letters, IEEE*, vol. 14, no. 12, pp. 1161–1163, 2010.

[137] C. Vaidyanathan and K. M. Buckley, "Performance analysis of the MVDR spatial spectrum estimator," *Signal Processing, IEEE Transactions on*, vol. 43, no. 6, pp. 1427–1437, 1995.

[138] V. Krishnaveni, T. Kesavamurthy, *et al.*, "Beamforming for direction-of-arrival (DOA) estimation: A survey," *International Journal of Computer Applications*, vol. 61, no. 11, pp. 4–11, 2013.

[139] Z. Szalay and L. Nagy, "Target modeling, antenna array design and conventional beamforming algorithms for radar target DOA estimation," in *Transparent Optical Networks (ICTON), 2015 17th International Conference on*, pp. 1–4, July 2015.

[140] N. Tayem, "2D DOA estimation of multiple coherent sources using a new antenna array configuration," in *Signals, Systems and Computers (ASILOMAR), 2012 Conference Record of the Forty Sixth Asilomar Conference on*, pp. 212–216, Nov 2012.

[141] F. Akbari, S. Moghaddam, and V. Vakili, "MUSIC and MVDR DOA estimation algorithms with higher resolution and accuracy," in *Telecommunications (IST), 2010 5th International Symposium on*, pp. 76–81, Dec 2010.

[142] J. Volakis, *Antenna Engineering Handbook, Fourth Edition*. McGraw-Hill Companies,Incorporated, 2007.

[143] J. M. Becker, *Dynamic beamforming optimization for anti-jamming and hardware fault recovery*. PhD thesis, CARNEGIE MELLON UNIVERSITY, 2014.

[144] L. Lei, X. Rongqing, and L. Gaopeng, "Robust adaptive beamforming based on generalized sidelobe cancellation," in *Radar, 2006. CIE'06. International Conference on*, pp. 1–4, IEEE, 2006.

[145] Y. Xu and Z. Liu, "Noncircularity-rate maximization: a new approach to adaptive blind beamforming," in *Wireless Communications, Networking and Mobile Computing, 2009. WiCom'09. 5th International Conference on*, pp. 1–4, IEEE, 2009.

[146] S. Chen, L. Hanzo, N. N. Ahmad, and A. Wolfgang, "Adaptive minimum bit error rate beamforming assisted receiver for QPSK wireless communication," *Digital Signal Processing*, vol. 15, no. 6, pp. 545–567, 2005.

[147] R. Haupt and H. Southall, "Experimental adaptive nulling with a genetic algorithm," *MICROWAVE JOURNAL-EUROGLOBAL EDITION-*, vol. 42, pp. 78–89, 1999.

[148] A. Massa, M. Donelli, F. G. De Natale, S. Caorsi, and A. Lommi, "Planar antenna array control with genetic algorithms and adaptive array theory," *Antennas and Propagation, IEEE Transactions on*, vol. 52, no. 11, pp. 2919–2924, 2004.

[149] Y.-j. Lee, J.-W. Seo, J.-K. Ha, and D.-c. Park, "Null steering of linear phased array antenna using genetic algorithm," in *Microwave Conference, 2009. APMC 2009. Asia Pacific*, pp. 2726–2729, IEEE, 2009.

[150] D. J. Sadler, "Planar array design for low ambiguity," in *Antennas & Propagation Conference, 2009. LAPC 2009. Loughborough*, pp. 713–716, IEEE, 2009.

[151] H. Evans, P. Gale, B. Aljibouri, E. Lim, E. Korolkwiwicz, and A. Sambell, "Application of simulated annealing to design of serial feed sequentially rotated 2× 2 antenna array," *Electronics Letters*, vol. 36, no. 24, pp. 1987–1988, 2000.

[152] S. Ram, *A Study of Adaptive Beamforming Techniques Using Smart Antenna For Mobile Communication.* PhD thesis, National Institute of Technology Rourkela, 2007.

[153] O. Bazan and M. Jaseemuddin, "A survey on mac protocols for wireless adhoc networks with beamforming antennas," *Communications Surveys & Tutorials, IEEE*, vol. 14, no. 2, pp. 216–239, 2012.

[154] A. AhmadAnsari, Z. Hasan, K. Mohammad Athar Siddique, and M. Jahangeer Alam, "Performance comparison for omnidirectional and directional MAC protocols for ad hoc network," *International Journal of Computer Applications*, vol. 70, no. 21, pp. 26–31, 2013.

[155] J. Niu, R. Zhang, L. Cai, and J. Yuan, "A fully-distributed directional-to-directional MAC protocol for mobile ad hoc networks," in *Computing, Networking and Communications (ICNC), 2015 International Conference on*, pp. 766–770, IEEE, 2015.

[156] M. S. Ullah, F. N. Nur, and N. N. Moon, "Optimization of wireless ad-hoc networks using an adjacent collaborative directional MAC (ACDM) protocol," *International Journal of Computer Applications*, vol. 114, no. 3, 2015.

[157] S. Bhunia, S. Sengupta, and F. Vázquez-Abad, "Performance analysis of cr-honeynet to prevent jamming attack through stochastic modeling," *Pervasive and Mobile Computing*, vol. 21, pp. 133–149, 2015.

[158] A. Nasipuri, S. Ye, J. You, and R. E. Hiromoto, "A MAC protocol for mobile ad hoc networks using directional antennas," in *Wireless Communications*

and Networking Confernce, 2000. WCNC. 2000 IEEE, vol. 3, pp. 1214–1219, IEEE, 2000.

[159] E. Troja, K. Ezirim, and S. Bhunia, "Route aware dynamic channel scheduling and selection for multi-hop cognitive radio networks," in *Vehicular Technology Conference (VTC Fall), 2013 IEEE 78th*, pp. 1–5, IEEE, 2013.

[160] W.-D. Wirth, *Radar techniques using array antennas (FEE radar, sonar, navigation & avionics series)*, vol. 10. IET, 2001.

[161] S. Chandran, *Adaptive antenna arrays: trends and applications*. Springer Science & Business Media, 2013.

[162] ns-3 Consortium, "Network simulator, ns-3." https://www.nsnam.org/.

[163] F. Bai and A. Helmy, "A survey of mobility models," *Wireless Ad hoc Networks. University of Southern California, USA*, vol. 206, 2004.

[164] O. Bouachir, A. Abrassart, F. Garcia, and N. Larrieu, "A mobility model for UAV ad hoc network," in *ICUAS 2014, International Conference on Unmanned Aircraft Systems*, pp. pp–383, 2014.

[165] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (AODV) routing," tech. rep., United States, 2003.

[166] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in *Proceedings of the Conference on Communications Architectures, Protocols and Applications*, SIGCOMM '94, (New York, NY, USA), pp. 234–244, ACM, 1994.

[167] "Directional antenna patch for ns-3." https://codereview.appspot.com/6620057/#ps1.

[168] M. Willerton and D. Yates, "Imperial College London Enhances Direction Finding and Beamforming Using LabVIEW and NI USRP." http://sine.ni.com/cs/app/doc/p/id/cs-15016#.

[169] "Angle of Arrival Detection with NI USRP RIO." http://forums.ni.com/t5/Software-Defined-Radio/Angle-of-Arrival-Detection-with-NI-USRP-and-LabVIEW/ta-p/3534214.

[170] P. A. Regis, S. Bhunia, and S. Sengupta, "Implementation of 3d obstacle compliant mobility models for uav networks in ns-3," in *Proceedings of the Workshop on Ns-3*, WNS3 '16, (New York, NY, USA), pp. 124–131, ACM, 2016.

[171] S. Bhunia, V. Behzadan, P. A. Regis, and S. Sengupta, "Performance of adaptive beam nulling in multihop ad-hoc networks under jamming," in *High Performance Computing and Communications (HPCC), 2015 IEEE 7th International Symposium on Cyberspace Safety and Security (CSS), 2015 IEEE 12th International Conferen on Embedded Software and Systems (ICESS), 2015 IEEE 17th International Conference on*, pp. 1236–1241, IEEE, 2015.

[172] S. Bhunia, V. Behzadan, P. A. Regis, and S. Sengupta, "Adaptive beam nulling in multihop ad hoc networks against a jammer in motion," *Computer Networks*, 2016.