2015

Synoptic analysis techniques for intrusion detection in wireless networks

Deanna Therese Hlavacek
Iowa State University

Synoptic analysis techniques for intrusion detection in wireless networks

by

Deanna Therese Hlavacek

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Computer Engineering

Iowa State University

Ames, Iowa

2015

DEDICATION

I would like to dedicate this dissertation to my husband Kenneth. Without his love, sacrifice, and support this accomplishment would not have been possible. I also dedicate this dissertation to my children Melanie, Douglas, Thomas, Amanda, and daughter-in-law Samantha for their unwavering emotional support - they are my biggest fans. I must include my granddaughter, Jadyn, for providing fun and recreation during my first year at Iowa State University by taking me to all of the playgrounds in Ames, Iowa and schooling me in the strategy of the game "Trouble". Dedication also goes to Dorothy and Eugene Hlavacek for providing babysitting and family support during my military career. Finally, I dedicate this dissertation to my parents Samuel and Shirley Craig. They instilled the love of life-long learning and the appreciation of a hard day's work. They are not here to see the conclusion of my efforts, but I know they are proud.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I want to take this opportunity to express my heartfelt thanks to those who helped me with my research, dissertation, and fulfillment of my publishing requirements. First and foremost I must offer profound thanks to Dr. J. Morris Chang for approaching me and initiating first contact in my first semester during his Wireless Network Architecture class. His guidance, demands to stay on schedule, inspirational chats, and valuable mentorship enabled me to achieve my goal in the three year time period I was allotted by my scholarship. I offer my deepest thanks to Dr. Doug Jacobson for taking a chance by allowing me to become the first PhD Cyber Corps Scholarship for Service recipient at Iowa State University. I also thank my committee members from the College of Engineering and the Department of Statistics for their support during this process, and for the inspiration and education I received in each of their classes: Dr. J. Morris Chang, Dr. Doug Jacobson, Dr. Yong Guan, Dr. Daji Qiao, and Dr. Bob Stephenson. Finally, I thank Dr. Michael Golemo, my committee member from the Department of Music, for welcoming me to Iowa State University and providing encouragement throughout my adventure.

CHAPTER 1. OVERVIEW

Wireless ad hoc networks are self-organizing and self-configuring infrastructure-less networks of nodes which are connected by wireless links. Examples are the 802.11/WiFi products and wireless sensor networks (WSNs). Wireless technologies also include mobile devices, such as cellular phones and the new cognitive radio network.

The popularity of these technologies is growing rapidly. Many of the tasks conducted on the devices require the use of Personal Identifiable Information (PII). The increasing frequency of incidents in which criminals are attempting to steal PII makes security of the information stored and transmitted from the devices, a well as the device itself, paramount.

Recently we have seen large scale attacks on networks that go initially undiscovered due to the large amount of data created by the intrusion detection tools themselves. High rates of false positives can have an effect upon system administrators that is similar to the effect on the townsmen in the story in which which the boy "cries wolf" too often. Additionally, large volumes of security data can create a situation in which the administrator does not have time to properly analyze it in a reasonable time. An example of this is the Target point-of-sale breach in 2013. Although an alert was sounded by the intrusion detection system FireEye, system administrators missed the warning. Over 40 million credit card numbers were sent to Russia before administrators investigated the alarm and took action to close the breach [107]. Similar breaches occurred at Home Depot and a variety of other retail centers. With high false positive rates for anomaly detection common, research has now moved more towards hybrid solutions. All of these methods require some prior training of the network nodes or pre-positioning of data for comparison.

Perhaps the solution is not more data. Rather, the solution may lie in the ability of a system to effectively use select data. Igor Baikalov, chief scientist at Securonix, was quoted in an article by the New York Times [107]: “We don’t need ’big data’. We need big information.” By carefully removing the noise created by large amounts of data, we allow our security professionals to focus on information with value, quickly identify attacks, and make timely decisions.

Our motivation for research related to intrusion detection arises from the current lack of comprehensive research into methods of analysis of selective information in an effort to construct a big picture of network security and integrity, termed as “network health”. Research into the parameters of the nodes and networks, the interplay of parameters and their effect upon each other, and how the concurrence of certain parameter levels portend negative or positive network health can bring valuable insight into the diagnosis of network ills.

Our hypothesis is that, by adapting a methodology borrowed from the science of meteorology, we can utilize the data available at both the node and cooperative network levels and create a synoptic picture of network health, providing indications of any intrusions or other network issues. Parameters such as packet delivery ratio, packet sequence number, route-add ratio, and many others have previously been used to alert on and/or identify intruders. However, this data also provides valuable information about the state of the network as a whole. By analyzing the packet, route, and node data at a network level we expect to develop a synoptic picture of the network. The visual representation of the synoptic network picture is expected to be much like synoptic weather charts depicting the pressure centers that, once properly analyzed, are indicative of changing weather. In this sense, just as barometers can monitor the environment for approaching storms, we have network tools that monitor parameters that can be used to identify malfunctioning areas of the wireless network. Since the synoptic analysis technique presented is founded upon comparing the counts of events in or effects on the wireless network it is anticipated that other attacks that are based upon causing or affecting countable events that trigger changes in network characteristics are candidates for synoptic analysis. Examples are attacks at the physical, network, and data link layers such as a sinkhole, jamming, and wormhole assaults.

Chapter Two consists of our first paper, “A Layered Approach to Cognitive Radio Network: A Survey”, which explores the attacks on the cognitive radio network [40]. In order to better understand the types of attacks levied on wireless systems, and the cognitive radio network in particular, we conducted a survey of the current research. By classifying the attacks according to the protocol layer at which the attack occurs we can better determine threat severity, precautionary methods, and recovery strategies. Additionally, basing the classifications upon protocol layers utilizes terminology already used in wireless communication security while simultaneously describing for the reader the attack vector. Finally, understanding the similarities between the threats can help us apply knowledge about previous attacks on other technologies to cognitive radio networks.

Our major contribution is presented Chapter Three in our second paper, "Design and Analysis of a Method for Synoptic Level Network Intrusion Detection" [41]. The result is a revolutionary way to analyze node and network data for patterns, dependence, and effects that indicate network issues. The paper describes a method based on utilizing packet delivery ratio (PDR), node reliability, route reliability, and entropy to develop a synoptic picture of the network health in the presence of a sinkhole.

Chapter Four presents our third paper, "A Method for Synoptic Level Intrusion Detection in a Wireless Ad Hoc Network", which revisits sinkhole detection and identification in a grid network. The work is expanded to include a sinkhole in a thirty node scrambled network, and a sinkhole in a one hundred node grid network. We also include the detection and identification of a HELLO Flood attacker using the same methodology.

CHAPTER 2. A LAYERED APPROACH TO COGNITIVE RADIO NETWORK SECURITY: A SURVEY

Modified from a paper published in *Computer Networks*[1]

Deanna T. Hlavacek[2][3][4] and J. Morris Chang[5]

2.1 Abstract

Cognitive radios have been identified as a solution to the crowded spectrum issue. With the realization of cognitive radio networks came the recognition that both new and old security threats are relevant. The cognitive radio network is still vulnerable to many of the denial of service, wormhole, routing, jamming attacks that plague other wireless technologies. In addition, the cognitive radio network is vulnerable to new attacks based on cognitive radio innovations, such as spectrum sharing, spectrum sensing, cognitive capability, and radio reconfigurability. The scope of this survey is to present an overview of security threats and challenges to the cognitive radio network, especially focusing on new solutions from 2012 and the first half of 2013. Included are prior mitigation techniques that are adaptive to the new technology, as well as new mitigation techniques specifically targeted at new cognitive radio vulnerabilities. The threats provided are organized according to the protocol layer at which the attack is targeted.

[1]Reprinted with permission of Computer Networks, 24 December 2014, Volume 75, Part A, Pages 414-436
[2]Graduate Student, Department of Electrical and Computer Engineering, Iowa State University.
[3]Primary Researcher and Author.
[4]Author for correspondence.
[5]Associate Professor Department of Electrical and Computer Engineering, Iowa State University.

2.2 Introduction

It has been estimated that the people of the United States are now outnumbered by their wireless devices. The proliferation of wireless devices such as laptops, notebooks, cellular phones, smart phones, and tablets has caused the frequency spectrum used for transfer of information to become crowded [84]. Also, the expected growth in media-rich consumer applications and wireless data transfer will continue to crowd the network, making additional spectrum throughput a priority.

Currently in the United States spectrum is allotted to various services in three main categories: licensed, lightly licensed, and unlicensed [1]. Licensed spectrum refers to the portions of the spectrum reserved by each country's equivalent of the Federal Communications Commission (FCC) for specific uses, such as military, public safety, and commercial uses. Lightly licensed spectrum refers to the bands that are generally regulated for licensed users, with regional or other exceptions. In the unlicensed band there are predefined technical rules for the hardware and radio technology intended to mitigate interference between the bands. The spectrum is available for network setup by any person or entity, public or private, to include commercial high speed internet, that does not infringe upon the band's rules [1].

In an effort to provide relief to the users of the overused spectrum, in 2010 the FCC allocated unused spectrum between television channels, or "white spaces" for unlicensed use. In addition, the FCC has proposed setting aside some low band spectrum, and possibly underutilized portions of the military, amateur radio, and paging frequencies, for unlicensed use as long as the primary user experiences no interference. Finally, in early 2013, the FCC opened a process to allocate more high frequency spectrum for unlicensed use.

Another of the major challenges for the wireless medium is security. The WiFi brand was adopted in 1999 based on the 802.11 standard. It was immediately realized that using the electromagnetic wave as the propagation medium made physical security of the transmitted data an impossibility. A conversation made of electromagnetic signals can be intercepted, jammed, or injected with extraneous bits. These actions can cause the release of private information, the inability to send and receive information, or the receipt of false or unreadable data.

As with other wireless communications, the cognitive radio technology based on the 802.22 standard must enforce the security triad of confidentiality, integrity, and availability (CIA). The cognitive radio is subject to many of the same types of attacks that plague other cellular and wireless communication systems. In addition, due to the cognitive radio's ability to self-organize a network and establish routing similar to wireless sensor networks (WSNs), the cognitive radio network (CRN) is also vulnerable to attacks originally designed for WSNs. Finally, the abilities of the CRN to sense the environment, adjust spectrum usage parameters, collaborate with neighbors, and learn provide new avenues for attack.

Because cognitive radio is in its infancy, there are many opportunities for research into the security issues to which the new technology is vulnerable. Such research can drive the creation of a more secure product. The papers [11], [69], and [83] provide a general overview of the cognitive radio network model with a broad description of secure model considerations. The authors of [68] provide a very extensive overview of all cognitive radio network issues, with an in-depth look at the security issues specific to the new CRN vulnerabilities.

The papers [52] and [106] each provide a high level view of the legacy and newer threats that can be applied to the cognitive network. The authors of [8] and [109] both take a broad stroke at listing and describing threats specific to the cognitive radio. In addition, the paper [8] adds a focus on the threats specific to the policy controlled cognitive radio. An in-depth look at the primary user emulation attack and mitigation is presented by the authors of [99] and [109]. The paper [88] analyzes vulnerabilities of existing spectrum sensing and access protocols under stochastic channels in the presence of jamming attacks. The authors of [92] concentrate on the vulnerabilities of the physical layer.

Comprehensive, security focused studies for the cognitive radio network were presented by [9, 32, 77] and [90]. The paper [90] takes the traditional approach of describing the possible attacks on a CRN. The authors of [9] categorize and analyze the threat vectors (as compared to attacks) and provides design considerations to alleviate the threats. A discussion on security evaluation and certification is included. Rather than analyzing the threats or attacks to the cognitive radio, the paper [32] analyzes the 2010 and earlier solutions presented to mitigate CRN security issues.

The paper [77] takes a layered approach in its study of cognitive radio network security. Four layers are presented: security applications, security strategies, security infrastructure, and security primitives. Threats are also presented in categories: learning, hidden node, policy, parameter, sensing.

The purpose of this paper is to provide a survey of security issues related to the cognitive radio network. Potential attacks will be described, and proposed mitigation techniques will be explored. The attacks in the survey are presented according to the targeted protocol layer. Emphasis has been placed on presenting solutions proposed in 2012 and early 2013, when available. The remainder of the paper is organized as follows: Section II describes the general concepts and security considerations of the cognitive radio. Starting at Section III the paper presents attacks and mitigation techniques based on communication layer protocols. Sections III through VII present the Physical Layer, Data Link Layer, Network Layer, Transport Layer, and Application Layer. Section VIII presents the Cross Layer attacks. The Cross Layer is a class of attacks launched at one layer with the intent to do damage to another layer. Section IX presents a framework for security of the cognitive radio network. Section X provides a conclusion. Table 2.1 will provide snapshots of the attacks presented by layer.

2.3 Cognitive Radio

The cognitive radio is based on a software defined radio with adjustable operational parameters [2]. The software allows the radio to tune to different frequencies, power levels, and modulation schemes to establish or maintain a communication link. The hardware consists of an antenna, a radio frequency conversion modules, a modem, and other modules [71]. The best configuration for the radio is determined by optimizing an objective function that considers such factors as interference and noise, traffic demand, mobility levels, and location.

In addition to the variable parameters mentioned above, the cognitive radio network is further adaptable to changing situations with its ability to operate successfully in collaborative (cooperative) or uncooperative networks. Generally, the throughput of the collaborative network will be higher than that of the uncooperative network due to the ability of the cooperating radios to share the frequency to which they will hop. However, when the network is under certain types of attacks, or in certain environmental situations, the uncooperative network configuration may be optimal. We must therefore analyze attacks and mitigation techniques for both scenarios.

It is generally agreed the cognitive radio must provide the following functions: spectrum sensing, spectrum management, spectrum sharing, and spectrum mobility. Spectrum sensing is required for the cognitive radio to sense the spectrum for the presence of the primary user or other traffic. Through spectrum management the radio is able to utilize the available spectrum efficiently without interfering with the primary user. The protocols established in the IEEE 802.22 standard govern the ability of the radio to share the spectrum with the primary user and other secondary users. The radio is able to vacate a spectrum when the primary user is indicated as present while continuing communication with the network due to the function of spectrum mobility. The spectrum functions required by the cognitive network radio add avenues of attack on the radio, network, and primary users in the area. These attacks may target the spectrum sensing function by changing the spectrum environment, the decision making function by manipulation of parameters of the objective function, or the learning engine by providing false data about the environment that the learning radio will use in the future to make incorrect or inefficient decisions.

By classifying threats we can better determine threat severity, precautionary methods, and recovery strategies. Additionally, understanding the similarities between the threats can help us apply knowledge about previous attacks on other technologies to cognitive radio networks. The following framework provides a classification system for all cognitive radio network threats. The threats are classified according to the protocol layer upon which the attack is performed: physical layer, data link (or MAC) layer, network layer, application layer, and cross layer. Cross layer attacks are those in which the attack is launched utilizing one layer while the attack targets another layer. Basing the classifications upon protocol layers utilizes terminology already used in wireless communication security while simultaneously describing for the reader the attack vector. We start our discussion at the bottom of the layer stack and move upwards.

Tables 2.1, 2.2, 2.4, 2.5, and 2.6 list each attack explored with the leg(s) of the security triad affected by the attack. A majority of the attacks affect the availability of the cognitive radio services. Protecting the system availability basically includes protecting the common control channel from saturation, ensuring the spectrum is sensed accurately, and the members of the network are properly identified and their information is accurate. Ensuring confidentiality and integrity of the data transmitted is accomplished by encryption with a proper key distribution system, and proper identification and vetting of the network members. Mitigation of the attacks listed will help ensure secure communications.

Table 2.1 Attacks By Layer - Physical

Attacks by Layer	**Network Member?**	**CIA**	**Description**	**Citation**
PHY Layer				
Jamming	External	A	Jammer maliciously sends packets to hinder legitimate spectrum usage.	[12, 14, 31, 60, 64, 75, 85, 93, 95, 97, 98, 111, 114, 119, 129]
Objective Function	Internal	A	Attacker manipulates transmission rate parameters so cognitive engine will calculate results that are biased towards the attacker's interests.	[7, 17, 21, 56, 96, 121, 124]
Overlapping Secondary User	Both	A	A geographical region may contain overlapping secondary networks with a malicious user in one network transmitting signals that cause harm to the primary and secondary users of both networks.	[115, 127]
Primary User Emulation	External	A	An external attacker emulates the signal of the primary user.	[16, 20, 27, 35, 36, 37, 48, 59, 65, 120, 126, 128, 132]

2.4 Physical Layer

The physical layer is the lowest layer of the protocol stack, providing an interface to the transmission medium. The physical layer consists of anything that is used to make two network devices communicate, such as the network cards, fiber, or, as in the cognitive radio network framework, the atmosphere. The operation of the cognitive radio network is more complicated than other wireless communication networks because the cognitive radio uses the frequency spectrum dynamically. Following are network attacks aimed at disrupting communication by targeting the physical layer of the cognitive radio network.

2.4.1 Primary User Emulator Attack

Testing results show that the number of dropped calls can be increased by up to two orders of magnitude due to primary user attacks [48]. Proper function of the spectrum sharing feature of the cognitive radio network requires the radio's ability to distinguish between the primary and secondary user signals. Techniques such as filter detection, energy detection, and cyclostationary-feature detection need to be leveraged to provide this distinction. In a hostile environment, distinguishing the primary user from others can become extremely difficult. In the primary emulation attack, an attacker may modify their air interface such that it emulates the primary-user's signal characteristics causing other secondary users to falsely determine the frequency is in use by the primary user, and so vacate the frequency. The imposter may perpetrate the attack selfishly, so he can use the spectrum, or maliciously, so the other legitimate users will have their communication disrupted, resulting in a Denial of Service attack. In addition, the attacker can poison the data collected about the spectrum usage that is used by the learning cognitive radio to determine which frequencies to try to access in the future. Therefore, the primary user attack (PUE) can lead to an objective function attack (section 2.4.2) [120].

Determination that there is an imposter present in the network is the first step in mitigating the PUE attack. This subject falls into the area of robust distributed cooperative sensing and the detection of anomalies. Most anomaly detection is based upon statistical analysis of the sensed data. Localization of the malicious user can assist in the mitigation of the attack.

The paper [65] provides a received signal strength indicator (RSSI) based transmitter localization technique that can be used when three or more trusted nodes are present. Triangulation with a correction technique considering multi-path signals and refraction provides an improved localization method.

In a cooperative cognitive radio network each secondary user senses the spectrum periodically and reports the measurement results to the fusion center. The fusion center combines the data and makes a determination as to whether the primary user is present or not. If an attacker injects false positive offset data, the fusion center may determine the primary user is transmitting, when actually it is not. Conversely, if the attacker injects negative data, the fusion center may falsely determine the primary user is not present.

In [36] a differential game is proposed as an avenue for primary user emulation mitigation. Based on the differential attack game model the Nash equilibrium is derived, and the optimal attack/defense strategy is devised. Experiment results indicate that using this strategy, the secondary user can maximize the usability of the cognitive channels, and minimize the disruption to the network due to primary user emulation attacks.

In the paper [59] the authors introduce the robust principal component analysis (PCA) technique for spectrum sensing. The authors consider a cooperative cognitive radio network with one primary user, several nodes, and one fusion center. In the worst case PUE attack, the attacker would use tactics that include appearing intermittently and randomly to try to prevent discovery. This activity can be represented by a sparse matrix. Robust PCA is based upon matrix theory and can be applied to get the estimated low rank matrix and the estimated sparse matrix from the corrupted observation matrix. Once the low rank and sparse matrices are estimated, the received signal power can be estimated for the suspect nodes. This transmission energy data is removed from the collected data at the fusion center. The data cache is no longer poisoned, and the determination of the presence of a primary user is more accurate.

The authors of [27] and [126] provide methods of determining if a primary user emulator is in the network when the primary user location is known and fixed. The method of the paper [27] is based on using a trust-based transmitter verification scheme to properly vet the primary user. It is assumed all radios are aware of the location of, and therefore the distance to, the primary

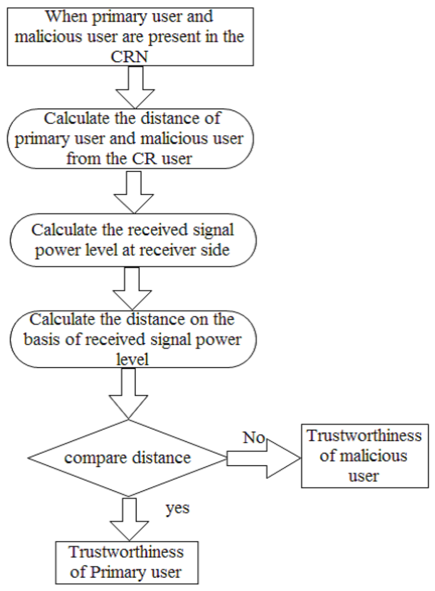
When primary user and malicious user are present in the CRN
Calculate the distance of primary user and malicious user from the CR user
Calculate the received signal power level at receiver side
Calculate the distance on the basis of received signal power level
compare distance
No
Trustworthiness of malicious user
yes
Trustworthiness of Primary user

Figure 2.1 Proposed transmitter verification scheme [27]

users in the area. The distance between the primary user and the cognitive radio is calculated based on known coordinates. The distance between radio and the user sending the primary user type signal is also calculated based on the received power levels. The trustworthiness of the user is determined by a comparison of the resulting distances. Figure 2.1 reflects the flow of the decision process.

In [126] the authors provide a method of defense against the primary user emulation attack using belief propagation. All secondary users in the network iteratively calculate the location function, a compatibility function, compute messages, exchange the messages with neighbors, and calculate the belief function until convergence. At convergence, any existing attacker will be detected, and secondary users will be notified of the attacker's signal characteristics via broadcast message. This allows all secondary users to avoid the attacker's primary emulation signal in the future.

The location function can locate the attacker based on differences in the received strength of the transmitted signal. Since none of the secondary users are aware of the transmitted signal strength or their distance from the attacker, the pinpointing of the attacker location depends upon the difference in measured signal strength by several neighbors. It was determined that one secondary node needs to interact with at least three neighboring nodes to estimate the attacker's location. After determining the location and computing the compatibility function until convergence, if the belief manipulation sum is higher than a specific threshold, the transmitter is determined to be the primary user, and not an attacker.

Similarly, the authors of [19] provide a transmitter verification scheme called LocDef (localization-based defense). The scheme verifies whether a signal is from an incumbent by estimating its location and observing the signal fingerprint. Localization is determined by utilizing an underlying wireless sensor network (WSN). The WSN collects snapshots of received signal strength across the cognitive radio network. The collected measurements are smoothed and the peaks are identified. Using the peaks, the transmitter locations can be identified.

The papers [16] and [35] propose methods for cooperative sensing in the presence of a primary user emulator and the probable detection of a primary user. When an attack is underway, secondary users in the area receive the signals from both primary user and attacker. This sensing information is sent to the fusion center. In [35], when differing signal energy is reported as determined by a network threshold, statistical probability is applied to the reports to determine if the primary user or a malicious emulator is present.

In [16] the information is combined with a weighting system to maximize the probability of detection within the constraints of a false alarm probability. The weights are related to the channel state information (CSI) between the nodes. The CSI is estimated using existing channel estimation algorithms. The method presented maximizes the probability of detection of the primary user by deriving optimal weights. It must be noted this paper assumes the primary user emulator (attacker) has been determined as present, and so the goal is to detect the primary user in the presence of the attacker with the assistance of multiple cooperative cognitive radio users.

Identification of a primary user emulator through a radio fingerprint has been proposed in the papers [20], [37], [117], and [132]. With a radio fingerprint, a wireless device can be identified by its unique transmission characteristics. Electronic fingerprinting is already used by cellular operators to prevent cell phone cloning. The fingerprint is due to the slight variations in the manufacture of the hardware components.

In the paper [20], the authors employ the spectrum sensing capability of the cognitive radio itself to identify primary user attacks. The uniqueness, or fingerprint, of the wireless signals is determined by use of the Neyman-Pearson test. The test is used to differentiate between the channel states of transmitters over Rayleigh fading channels. Simulation showed the method was effective in identifying a primary user emulator, thereby allowing the network to defend against the attack.

The authors of [132] focus on the phase noise of a signal created by the local oscillator. Phase noise is the rapid, random fluctuations in the phase of the waveform. It causes spectrum spread and deformation, and is unique. After extraction of the phase noise from the received modulated signal, applet wavelet and higher-order statistical analysis is applied to identify the fake primary user transmitters. Results of simulation experiments showed the phase noise of two receivers using the same local oscillators was different. This indicates it is feasible to identify a transmitter for primary user emulation defense.

Performance analysis of the cognitive radio network is the focus of the paper [128]. The authors create a three dimensional Markov model to provide a method of performance analysis using a common control channel when under primary user attack. The outage probability metric is redefined, and the new performance metric common control channel recovery time is introduced. Together, the metrics identify and evaluate the impact of the common control channel on the network. The blocking rate and dropping rate of the cognitive radio network are also calculated.

Outage probability, or the probability of network suspension, reflects the chance a cognitive radio network will suspend. Suspension occurs with the arrival of a new primary user when all of the available N channels are already being utilized by only primary users, fake primary users, and the common control channel. Since no secondary users are currently using a channel, the common control channel must drop, opening the frequency for the new primary user. At this point the cognitive radio network is suspended.

With the system state defined as $(i,\ j,\ k)$ where i and j represent the number of primary users and primary user emulators, and k represents the sum of the secondary users plus the common control channel, the outage probability is the sum of the state probabilities where $k =$

0. Therefore, the outage probability is determined by

$$P_{outage} = \sum_{(i,\ j,\ k)\epsilon\Omega_1} P_{(i,\ j,\ k)}\ , \qquad \text{(Eq. 1)}$$

where $\Omega_1 = \{(i,\ j,\ k)\ /\ i + j = N\ and\ k = 0\}$.

The common control channel recovery time is the average time expected for recovery after an outage. The common control channel will only recover by using a channel vacated by a primary user or primary user emulator. The analysis is based upon the property that the sum of two Poisson processes results in a Poisson process. Therefore, the state probability distribution combined with the holding time of the local primary users and emulators provides the common control channel recovery time as

$$T_{\text{ccc}} = \sum_{(i\ j,k)\epsilon\Omega_2} \frac{1}{i\mu_{PU} + j\mu_{PU} + \lambda_{PU}} P_{(i,j,k)} + \frac{1}{N\mu_{PU}} P_{(N,0,0)}, \qquad \text{(Eq. 2)}$$

where $\Omega_2 = \{i + j = N,\ j > 0,\ k = 0\}$, the arrival rates of PUs, fake-PUs and SUs are λ_{PU}, λ_{fPU} and λ_{SU}, respectively, and the channel holding times are exponentially distributed with the mean, $\frac{1}{\mu_{PU}}$, $\frac{1}{\mu_{fPU}}$, and $\frac{1}{\mu_{SU}}$.

As expected, the analysis of tests using the formulas above show that a network under the attack of primary user emulators takes a longer time to recover than a network not under attack. This recovery time was also shown to increase as the number of primary user emulators increased. Additionally, the larger the number of primary user emulators, the greater chance the network would drop into the suspended state.

2.4.2 Objective Function Attack

Cognitive radios are adaptive to the environment. Many radio parameters are available for manipulation in the effort to adapt the radio to the environment by maximizing objective functions, and therefore the radio's ability to communicate over the medium. Objective function attacks apply to an attack on any learning algorithms that utilize objective functions. Another name for objective function attacks is belief-manipulation attacks. Parameters manipulated include, but are not limited to, bandwidth, power, modulation, coding rate, frequency, frame size, encryption type, channel access protocol.

The authors of [21] give the following objective function example. Assume the function exists where w_i are weights, P is power, R is rate, and S is security.

$$f = w_1P + w_2R + w_3S \qquad \text{(Eq. 3)}$$

Now assume an attacker wishes to lower the security with which the radio is transmitting messages. The attacker would monitor the channel, and jam the channel whenever the radio tries to send a message at the more secure level. The cognitive radio would learn that attempting to transmit at the higher security level would not be successful. This would result in either the higher security messages being sent at a lower security level, or the messages would not be sent at all. Similar attacks could cause a radio to avoid certain frequencies, rates, modulations, or bandwidths.

There have been few clearly effective methods of mitigating objective function attacks. One simple proposal has been made by [56]. The proposal suggests naively defining thresholds for each of the adjustable parameters. Communication would be prevented when one or more of the parameters did not fulfill its predefined threshold.

The authors of the papers [17] and [124] present the covert adaptive injection attack. In these examples of an objective function attack, the attacker is capable of learning and adjusting its strategies in response to the environment. The attacker attempts to stealthily manipulate the sensing results of a distributed network, thereby attacking the objective functions and decision making of the cognitive radio network. A robust distributed outlier detection scheme is presented to counter the covert attack.

The method presented by [124] uses a localized detection threshold at each node, and adapts the threshold with the diminishing behavior of state differences, exploiting the state convergence property. With this scheme, it is more difficult for an attacker to guess all of the thresholds of the neighbors at any instance. When a network node suspects an attacker, it sends a primitive alarm to its the immediate neighbors. The alarm is not forwarded. If the node collects primitive alarms from at least half of the nodes that are common neighbors of the node and suspected attacker, it broadcasts a confirmed alarm. The confirmed alarm is forwarded to the remaining network. Verification of the attacker is provided using a hash-based computation. This verification ensures the correctness of a neighbor's state update process with the goal of thwarting collusion attacks by common neighbor cross-validation.

Alternatively, the method presented in [17] uses a neighborhood voting system. After each secondary user has collected the sensing reports from its immediate neighbors, the nodes determine an algorithm based mean, and conduct a spatial correlation test. Based on the results, each node casts votes about the legitimacy of each of its neighbors. If a node receives more than half of the neighbor votes categorizing it as suspicious, the node is considered malicious.

The authors of [96] present a solution to the false channel information exchange attack. This is a form of the objective function attack because the goal of the attacker is to affect the decision making algorithms of the network nodes. The authenticity of the received channel information is analyzed using spatial correlation algorithms. Simulation shows that the algorithms achieve a high detection rate of malicious nodes with a low false alarm rate.

In [121] the authors explore a framework of power control schemes based on a robust Markov decision process. If an attacker can influence the power scheme of the radio, the attacker can affect the throughput of the network. Additionally, the authors use a delayed Markov decision process to model the throughput maximization problem while experiencing spectrum sensing delay caused by a malicious user. The delayed Markov decision process is solved by using a modified dynamic programming approach.

Belief manipulation attacks as related to the knowledge base of learning algorithms is presented in [7]. Many defense methods have been studied as related to the mitigation of jamming and other throughput affecting attacks. However, less studied has been the effect on the learning that takes place over time based on the objective function results, and how the learning is poisoned by intermittent attacks. To determine if there is an attacker present, monitor nodes are assigned to sample the channels over a time window and Wald's Sequential Probability Ratio Test rule is applied.

The softmax policy [100] includes randomized user actions based on some probability distribution in an effort to hide information about the learning algorithm. In the algorithm, more weight is applied to actions that performed well in the past. By avoiding the attacker's influence by using and sensing channels where the attacker is not expected to be present, the learning algorithm is reinforced and becomes increasingly accurate.

2.4.3 Overlapping Secondary User

As shown in Figure 2.2, a geographical region may contain coexisting, overlapping multiple secondary networks. Such a situation places dynamic spectrum access sharing at risk through both objective function and primary user vulnerabilities by one malicious node, or accidentally by a friendly node. A malicious user in one network may transmit signals that cause harm to the primary and secondary users of both networks. Signals transmitted maliciously may provide false sensing information, thereby negatively affecting the objective function in one or both networks. The malicious user may intermittently falsely emulate the primary users of each network causing each network to vacate the channel. Additionally, in special situations, a friendly node reporting the presence of the primary user in network one may inadvertently be relaying the same information to network two, negatively impacting network two's objective function. This attack can be hard to prevent since the malicious node may not be under the direct control of the secondary station or users of the victim network. This is essentially an attack on the capability of the cognitive radio network for spectrum sensing and sharing of both infrastructure and ad hoc based networks. The result is a denial of service attack.

The authors of [127] provide three possible mitigation solution categories for the overlapping secondary user attack. These mitigation techniques are also applicable to many other denial of service attacks, and are based upon work in other areas.

1. **Modifying the modulation scheme**: The use of frequency hopping and direct-sequence spread spectrum techniques can make it more difficult to launch effective denial of service attacks. The attacks may still degrade service quality.

2. **Detection and prevention of attacks**: Observing the primary user's location and signal characteristics, as described in section 2.4.1, "Primary User Emulator Attack", can help the network identify if a node is performing maliciously.

3. **Using authentication and trust models**: In the paper [115] a system is designed to determine a suspicion level, trust value, and consistency value to identify and exclude a malicious user. Nodes become suspicious when the reported channel state is not in agreement with the channel state reported by others. A trust value for each node is calculated over time, and a consistency value reflects the consistent trust value over time. A node with a consistently low trust value will eventually be identified as a possible malicious user and dropped from the network.

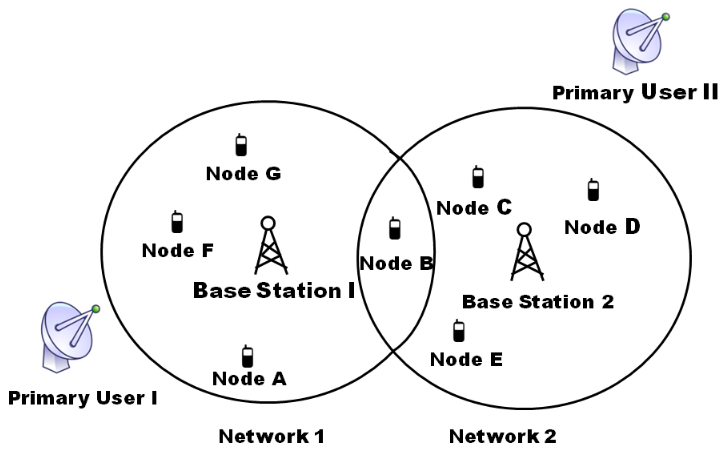

Figure 2.2 Overlapping Secondary Attack

2.4.4 Jamming

Cognitive radio networks require a minimum signal-to-noise ratio to decode a signal sent from their corresponding transceivers. Jamming, one of the most basic types of attacks in the cognitive radio network, attempts to adversely affect the signal-to-noise ratio. In this attack, the malicious user intentionally and continuously transmits on a licensed band, making it unusable by the primary or other secondary users. The attack is amplified by transmitting with high power in several spectral bands. Jamming can be detected with triangulation and energy based techniques. However, the time lost with these techniques allows the attacker to severely impact the network. A mobile attacker can be even more difficult to locate.

Before initiating mitigation techniques against jamming, the cognitive radio network must first determine that a jammer exists. Besides the presence and actions of a jammer, poor performance experienced by a receiving node can also be caused by natural causes such as network congestion.

A statistical approach is often used for detecting anomalous spectrum usage attacks, specifically stealthy jamming, and is proposed in both papers [85] and [97]. In [85], the statistical analysis is a three-step cross-layer process. First, statistical analysis is performed on the information gathered from multiple layers. Next, a multiple layer discrepancy search is conducted on the data collected by comparing the data from several layers. In the third step, simple statistical measures are used to determine if there are discrepancies among the data from the network and physical layers using only snapshot data. For instance, the physical layer may report numerous available channels in the area, but few nodes appear in the resultant paths. This may indicate jamming is occurring. Due to the possibility that there can be other reasons the nodes do not appear, there could be a high false alarm rate if a comparison to historic data is not conducted.

Using time series data available from multiple layers can minimize the false alarm probability. This is because the probability distribution of chosen observables will change when the network is under attack. The observables are carefully chosen such that their statistics will indicate a sharp change with high probability in the presence of an attacker. Although it is assumed the data from different layers is independent, it has been shown that the observed changes before and after the event are related via time.

In the paper [97] sequential detection is used to compare the statistical distribution before and after an attack. Confirmation of attack is obtained by a cross-layer three-step process. First, the statistical analysis of the paths/nodes is obtained from route discovery. If there are anomalous patterns observed, passive checking is performed by cross checking the pattern with the physical layer spectrum sensing results. Last, active checking is performed by selectively injecting controlled traffic into the potentially congested area and collecting measurements. The passive and active steps are conducted to confirm the results in the statistical analysis.

Jamming is an attack that affects both cooperative and uncooperative cognitive radio networks. In general, uncooperative networks are more resistant to jamming attacks because the nodes do not need to use a common channel to share information about the frequency to which they are hopping. In cooperative networks, the jammer can either capture the shared channel information and move to the same frequency to continue the attack, or inhibit the channel data exchange by jamming the common channel. However, although existing anti-jamming schemes for uncooperative networks are more robust when under attack, they are not as efficient as cooperative network channel sharing schemes when not under attack [98]. With no jammer present, network throughput is lower in uncooperative networks because the nodes need to use energy in the attempt to discover upon which channel the intended transmitter/receiver is transmitting/listening. Therefore, combining cooperative frequency sharing techniques with uncooperative networking and anti-jamming methods will make the cognitive radio network adaptable to changing network conditions while preserving network throughput. Below we describe anti-jamming methods for both cooperative and uncooperative networks.

2.4.4.1 Cooperative Network Jamming Mitigation Techniques

A scenario comprised of a primary user, secondary user, and jammer was studied in the paper [12]. The authors conducted a simulated jamming attack to derive the best combinations of the number of control and data channels to enhance the legitimate secondary user transmission during jamming. The data and control channel allocation determination was also specific to the type of application and the quality of service required for good throughput of the application. It was shown that there is a tradeoff between efficiency and transmission probability when allocating more than one channel to common control. Additionally, it was noted that the results did not always conform to what was initially expected. For an example, using the extremely conservative strategy of five control channels and three data channels was less efficient than using a less conservative strategy of four control channels and four data channels for an electronic mail application under jamming attack.

The paper [114] explores collaborative defense of the network against collaborative jammers. The collaborative defense is mounted using a multi-tier proxy-based cooperative defense strategy designed to exploit the temporal and spatial diversity available to the legitimate users in an infrastructure-based cognitive radio network. The network is divided between proxies and followers. The proxies act as relays between the followers and the base station. Followers must connect to a proxy, rather than straight to the base station. This adds another layer to the communication hierarchy. When the users cooperate, the jammers necessarily need to jam both the followers and proxies to jam all communication. Therefore, with the collaborative defense strategy, the jammers need more jammers to effectively suspend the network communication. Simulation results show that spectrum availability is greatly improved when the users cooperate. However, due to the extra layer in the communication hierarchy, the latency of communication is also increased.

A targeted jamming attack and its mitigation is presented in [43]. The authors describe the "Most Active Band" attack in which a jammer determines and targets the band with the most traffic for jamming, resulting in denial of service on that band. The coordinated concealment strategy (CCS) is offered as a countermeasure. Basically, a few secondary user nodes sacrifice themselves by moving to a single band, drawing the attacker's attention. The "surviving" nodes are free to operate on other bands under the concealment of the ruse.

In [93] and [111] the authors assume the jammer's signal and the primary user signal are distinguishable and the attackers will not jam the primary user. The contention between the jammer and the secondary users is based upon the secondary users' aim at maximizing spectrum utilization with carefully-designed channel switching schedules, while the malicious attacker's desire is to decrease spectrum utilization by strategic jamming. From this description, the objectives of the secondary user and jammer are opposite, and can be modeled as a zero-sum game. In the game model the secondary users adapt their strategy on switching between control and data channels according to their observations about spectrum availability, channel quality, and attacker's actions. According to simulation the calculated optimal policy can achieve better performance in terms of throughput as compared to a learning policy that only maximizes the payoff at each stage while not considering the environment dynamics, the attackers' cognitive capability, and a random defense policy.

A game-theoretic perspective is also used to determine the optimal defense strategy in [129]. A simple stochastic swarm optimization algorithm, called particle swarm optimization (PSO), is applied to solve the optimization problems numerically. PSO is motivated by many natural phenomena, and has been shown to represent each group member seeking the optimal solution for itself as it relates to its neighbors.

2.4.4.2 Uncooperative Network Jamming Mitigation Techniques

The authors of [23] provide a jamming solution based on a distributed, probabilistic protocol. This method is unique in that it avoids control channels, does not require information related to the node neighborhood, and does not require statistics about the channel usage. The solution is based upon probabilistic pairing approach that allows the node to dynamically find a peer and sync on a random, available frequency. The solution also requires the nodes be preloaded with pairwise keys which are used as seeds to the process of finding a common frequency band. In the syncing process, each node randomly chooses a key and challenges its neighbors. If there is a collision, the nodes agree on a frequency band for communication. Nodes experiencing no collision again randomly choose a key and challenge their neighbors. For the scheme to work, each node needs to dynamically sense the spectrum to determine a frequency free for use.

The authors of [119] developed a channel hopping defense strategy using the Markov decision process approach based on a secondary user that uses only one channel. To adequately use the decision process the user must learn some attacker information by observing the environment. The secondary user first estimates useful parameters based on past observations using maximum likelihood estimation (MLE). The user then utilizes the Q-learning process, which is presented as an avenue for the secondary user to learn and update the defense strategy without knowledge of the underlying Markov model. The scenario is extended such that the secondary user can utilize all available channels simultaneously. In this scenario, randomized power allocation is used as the defense strategy. Derivation of the Nash equilibrium for this Colonel Blotto game provides minimization of the worst-case damage.

In the paper [18], the authors developed a similar jamming-hopping, policy iteration scheme based on the Markov Decision Process which utilizes the Q-learning process to lessen the computation burden. However, in this scheme the secondary user has a finite set of channels from which to choose. The set of channel choices is dependent upon the state of the environment at decision time.

The paper [31] uses a game-theoretic context to formulate the interaction between communicating nodes and an adversary. Experimental results show that randomized actions by both the secondary user and the jammer result in lower game values than the expected Nash equilibrium for pure information-centric channel capacity. Results also show that packetized, adaptive communication is an advantage for the power-limited jammer. Additionally, it is proven there exists a threshold on the average power of a jammer above which the transmitter must use a rate equivalent to the maximum power of the jammer.

The authors of the paper [95] present the solution to jamming modeled after a solution to a multi-armed bandit problem. In this scenario the secondary user is the player trying to pull the most rewarding lever at each time slot. The authors use Whittle's linear program to determine which channel the secondary user should select for transmission. The model is valid for a situation in which the state of the non-accessed channels changes when not chosen. For the situation in which the state of the channels is static even when not chosen (in other words, the jammer's strategy is fixed), the author's solution is based on a stochastic multi-armed bandit process using indexing solutions.

Similarly, the authors of [112] and [113] formulate the jamming problem as a multi-armed bandit problem. In this solution the secondary sender and the receiver both adaptively choose their sending and receiving channels by basing their decisions on all of their past decisions and observations. With the convergence of the learning algorithms, the sender and receiver hop to the same set of channels with high probability under the presence of a jammer.

The paper [98] presents the Uncoordinated Frequency Hopping (UFH) scheme in an effort to allow key establishment between two nodes in the presence of a jammer without a pre-shared key. With the assumption that a jammer cannot jam all of the communication channels at the same time, the message is divided into multiple parts and sent across several frequencies according to a random frequency hopping scheme. Although a secret channel sequence is not utilized by the sender and receiver, it is shown that with sufficient transmission attempts the sender and receiver will converge upon the same channels in a number of time slots. Note also that the time slots for the sender and receiver do not need to be synchronized; instead the receiver is allowed to switch channels less often than the sender. The effect is a reduced number of partially received fragments. Experimental results show that the UFH scheme achieves the same level of anti-jamming protection as coordinated frequency hopping. However, the experiments also show that the UFH scheme results in lower communication throughput with higher storage and processing costs.

The authors of [61] present the time delayed broadcast scheme (TDBS). The scheme does not rely upon commonly shared secrets or common control channels to coordinate broadcasts. Alternately, the scheme relies upon a pseudo-noise (PN) frequency hopping sequence to establish communication. Unlike conventional PN sequences for multi-access systems, the PN sequence presented exhibits high correlation to enable broadcast. Additionally, the experimental results show the TDBS scheme can support and maintain broadcast communications while in the presence of an inside jammer.

The paper [75] examines the resiliency of rate adaptation algorithms (RAA) against smart jamming attacks. According to the experimental results, several techniques can prevent smart jamming by limiting the amount of key information that can be inferred by an attacker. The lack of information forces the attacker to operate as a memory-less jammer. For example, the SampleRate protocol can be protected by using randomized, non-sequential probing. To conceal the explicit and implicit rate information, such information should be protected using post-coding encryption. Using a shared secret key and a random initialization vector can ensure the explicit and implicit rate information is concealed.

In the work [14] the authors present another secret-sharing mechanism that does not require pre-shared secret keys. The method is called Time Reversed Message Extraction and Key Scheduling (TREKS). The TREKS mechanism is shown to be efficient and adversary-resilient and is based upon intractable forward decoding and efficient backdoor decoding. As with the other methods provided which do not use pre-shared secret keys, TREKS solves the circular dependency problem. Additionally, experimentation showed that TREKS was four magnitudes faster than the prior solutions to CDP, with minimum storage overhead and at most twice the computation required for traditional spread spectrum communication.

2.5 Media Access Control Layer

The Media Access Control (MAC) layer is a sublayer of the data link layer. The MAC layer is designed to support multiple users on a shared medium within the same network. A Common

Table 2.2 Attacks By Layer - Data Link

Attacks by Layer	**Network Member?**	**CIA**	**Description**	**Citation**
Data Link Layer				
Byzantine	Internal	A	Attacker sends false local spectrum sensing results to neighbors/fusion center causing the receiver to make wrong spectrum sensing decisions.	[4, 25, 26, 27, 30, 38, 45, 50, 62, 70, 74, 76, 79, 80, 81, 87, 123]
Control Channel Jamming	Both	A	Jamming of the control channel causes network confusion by interrupting the radio cooperation.	[15, 54, 61, 63, 64, 66, 101, 102, 130]
Control Channel Saturation	Internal	A	Based on the fact that if a cognitive radio is unable to complete negotiations during the limited time of the control phase, the radio defers from transmission during the next data phase.	[66, 73]

Control Channel (CCC) may be used for an exchange of control messages to coordinate the users.

2.5.1 Byzantine Attack

In the Byzantine attack, also known as spectrum sensing data falsification, the attacker injecting the false sensing information into the decision stream is a legitimate member of the network, and is referred to as the Byzantine. Byzantines may perpetrate the attack to selfishly acquire increased spectrum availability for themselves, or the attackers may have a goal of disrupting the throughput of the network for other nefarious reasons.

The authors of [103] propose a method of detection of Byzantines called Pinokio. Pinokio uses a Misbehavior Detection System (MDS) that maintains a profile of the network's normal behavior based on training data. The MDS detects misbehavior by monitoring the bit rate behavior. By protocol, the bit rate should change periodically, should be adjusted by a node contiguously, the bit rates between two nodes should show some reciprocity, and usage of a low-bit rate over a wide channel. Nodes exhibiting these characteristics are not acting in a manner conducive to spectrum efficiency, and so are suspect.

Another method of misbehavior detection called Cooperative neighboring cognitive radio nodes (COOPON) is provided by the authors of [49]. Detection of the selfish node is detected by the cooperation of other legitimate neighboring nodes. All of the secondary nodes exchange channel allocation information both received and sent to the suspect nodes. Each neighbor compares the number of channels reported to be used by the suspect node to the channels the neighbors report as being used. A discrepancy reveals a selfish actor.

Several techniques have been proposed related to trust and reputation metrics. In the context of cognitive radio networks, trust and reputation based schemes are very similar. Trust in a behavior based model is defined as the mutual relationship between two entities for a specific action. Trust most often refers to acknowledging nodes that are proven trustworthy in some way. Alternatively, reputation schemes are generally more interested in identifying those nodes that are bad actors.

A trust framework is proposed by [70] consisting of a TrustPolicy Engine and a TrustMetrics Engine. The TrustPolicy Engine targets four main areas: security mechanisms, enhancing secondary user's worthiness, spectrum sharing with legacy networks, and inter-operator spectrum sharing. The engine is designed to analyze the behavior of a cognitive radio according to the notion of trustworthiness. The proposed solution considers two types of trust. Social trust is based on historical actions; quality of service trust is related to performance issues. Both trust aspects are used to determine a trust level based on past behavior and impact on the performance of the network.

The TrustMetrics Engine provides for the exchange of trust information between the nodes and algorithms. This information consists of past performance actions and impacts, and is used to create a prediction of future behaviors. The data is passed to a Performance Engine that implements mechanisms to thwart cognitive radio network attacks.

Several recent authors have tackled the idea of trust or reputation based mitigation methods for the Byzantine attack using the sensed data sent to the fusion center. The authors analyze the case in which the Byzantines do not send true information about the state of spectrum. The information sent by the suspect nodes is compared to the information received from a trusted node.

In the paper [30], a node's reported sensed data that deviates from the data supplied by a trusted source results in the node being labeled as malicious. In [74], when differing signal energy is reported as determined by a network threshold, statistical probability is applied to the reports to determine if a malicious node is present.

The papers [25], [62], [87] and [122] all use reputation-based detection schemes over time to identify bad actors. In these schemes, a reputation measure is assigned to each node representing the number of times the local decision of a node was different than the global decision of the fusion center in a time window. The higher the value of the measure, the less reliable the node's observation is considered. To increase the accuracy of the decisions made by the fusion center, data from nodes with a high number of mismatches is not included in the sensing algorithms. The papers differ in the algorithms, weights, and observables used to determine the trust levels of the nodes.

In [76], [79], and [80] the authors present trust based authentication systems. In the model in the paper [76], the node with the highest level of trust is appointed as the base station. Authentication between each node and the base is accomplished in two ways. With cryptographic authentication, the base station generates secret keys for the network members. Each node shares a unique key with the base station.

With the certificate based trust authentication technique, the base station generates trust values for each node. Trust values are based upon the recent activities of a node in the network, such as success or failure in forwarding a packet, and the length of time the node has been a member of the network. The trust value of each node is updated by the base station every time the base station sends a broadcast message. If a request is received by the base station, the base station references the node's trust value for a determination of the appropriate action. A second secret key is shared between the base and all nodes of a specific, equal trust level.

The base station also sets a trust threshold for the network. Any node not meeting the minimum trust threshold is expelled from the network and placed on a blacklist for rejection of future joining or other requests. By using the trust method, any bad actors, or Byzantines, in the network will be identified and segregated from the network.

The method to assign trust in [80] is based on three factors for determining sensing trust level. Context is the first factor and includes time, location, spectrum, code, and angle. The second factor is based on sensing evidence scope and importance. This factor reflects the importance of evidence based on the impact of the action on the network. Lastly, all node behavior is collected relative to a time window. The time window allows a node fluctuating between trustworthy/untrustworthy and considerate/inconsiderate behavior to be properly analyzed for intent over time. Together the three factors help capture the transition of a benevolent, well-behaving node to a malevolent node over time, allowing the network to properly and continuously identify currently misbehaving nodes. Additionally, the algorithm allows the node's reputation to rise slowly but fall quickly to punish a secondary user's erratic behavior. The reputation values are considered in data fusion and resource allocation for the secondary users.

The trust calculation presented in [79] relies on several steps and inputs. The direct trust calculation is based on a cumulative attribute determined by the success or failure of past requests, responses, and retransmissions. The indirect trust calculation considers the neighbors' determination of the node's trust. The trust values are integrated, and a historical trust value is added to the algorithm. The node's ability to access the network resources is based upon the trust determination.

The unfair penalization of honest users due to severe pathloss in some locations is considered in the trust based scheme proposed by [45]. The proposed Location Reliability and Malicious Intention (LRMI) trust metric has two parts:

1. Location Reliability reflects pathloss characteristics of the wireless channel.

2. Malicious Intention captures the true intention of secondary users.

Evaluation of sensing reports sent to the fusion center is based on two sources of evidence - the cell the report was sent from (Location Reliability) and who generated the report (Malicious

Intention). A trust value is applied to each cell based on the activity of the cell members. The Dempster-Shafer theory is used to evaluate trustworthiness as related to a mobile node. The algorithmic combination of the two values help to alleviate the trust devaluation that generally occurs due to a node's signal pathloss because of its location and mobility, hence providing a more accurate trust determination.

In the paper [38] the authors present an alternate detection method using two conditional frequency check statistics (CFC). The statistics are developed under the Markovian model for the spectrum state and are not adversely affected by an increasing number of Byzantines. The newly proposed CFC enforces two constraints on the attacker's behavior as compared to the conventional one constraint. This is done by exploring the correlation between the consecutive spectrum states.

The fusion center evaluates the two CFCs for every sensor and compares the results to those of a trusted sensor. Differing values between a sensor and the trusted sensor indicate the corresponding sensor is malicious. Consequently, any flipping attacker that maliciously flips its local inference can easily be identified with the CFC. With at least one trusted user the method can achieve an accuracy rate of greater than 94% in detecting malicious users.

Statistically based analysis schemes that detect malicious users and alleviate the false sensing observations are proposed in [4] and [50]. The first scheme, proposed by [4], allows for an unknown number of malicious cognitive radios in a network, with the possibility that any node can suddenly turn malicious. The mathematical basis for sensed data analysis is a modified version of the Grubb's test for the detection of a single outlier in a normally distributed data set. Simulations showed that the modified Grubb's test was able to detect any number of malicious cognitive radios in a network, as long as at least half of the network was made up of trustworthy nodes. The second paper [50] compares the Dixon's test for outliers, the Grubb's test for outliers, and the box plot test when applied to sensed data. It is shown that the Dixon's test out performs the Grubb's test and the box plot test in detecting the presence of a single bad actor.

The authors of the paper [81] use a statistical attack model to aid in the development of a Bayesian approach to identifying malicious nodes. Belief propagation is used with factor graphs to solve the Bayesian estimation problem and the derivation of an algorithm. The algorithm is used to estimate channel status and the attack probabilities of the malicious nodes, thereby identifying the Byzantines.

A technique using the primary user's received signal strength (RSS) is introduced in [123]. The method has been shown to work no matter the ratio of trustworthy nodes to malicious nodes in the network. The technique compares the location determined by the strength of the primary user's received signal at a secondary user and reported to the fusion center, to that calculated using the combined data from the network secondary users at the fusion center. This comparison is used to determine whether the secondary user node is providing true or false data.

The authors of [26] present a punishment based mitigation scheme. Using the indirect punishment method, the malicious user does not need to be identified. There only needs to occur collisions with the primary user. It is assumed that when such a collision occurs, the primary user applies a punishment to the entire network. If the attacker can be determined, a punishment is applied directly. Assuming the bad actor is acting selfishly, either punishment will deny the malicious node throughput over the network, and will cause the node to change its behavior.

Alternatively, the authors of [10] present an incentive, or payment based solution that makes it detrimental to a node to refuse to forward packets over free channels. The basis of the system is that a node will receive payment after offering a free channel to forward packets for a neighbor. A transmitting node will pay a neighbor for packet transmission over a channel when that neighbor's services are required for transmission. A central authority is required to maintain the credit balance for each node.

2.5.2 Control Channel Saturation

The control channel saturation attack can occur if a cognitive radio is unable to complete negotiations during the timeframe of the control phase, the radio defers transmission during the next data phase. This situation may occur when the channel is saturated by a large number of contending cognitive radios. An attacker can broadcast a large number of packets with the intent to saturate the control channel. By sending different types of packets, a malicious node reduces the risk of detection. Combining the control channel saturation attack with the small window backoff attack (described in section 2.9.2 “Small Backoff Window”), the attacker may be able to ensure the malicious node captures the control channel before other users.

The authors of [66] propose using dynamic channelization to address the common control channel access problem. The authors define an atomic channel as a basic unit of b Hz. Upon the event of control channel migration, a composite channel is formed from the atomic channels, centered around a new carrier frequency. The formula $f = f_0 + mb$ provides for the shifting of the center of frequency from f_0 to f by a multiple of the basic unit b Hz where $m = 0, \pm 1, \pm 2$, etc. The bandwidth around f can be obtained by channelization as a factor of kb, such that $k = 1, 3, 5$, etc. Figure 2.3 shows the migrated control channel for the case of $(m, k) = (4, 3)$.

The paper [73] presents a method to react to control channel saturation with an alternative decision making strategy based on rendezvous negotiation to ensure user’s communication co-ordination. In essence, the paper presents an mathematical analysis of the resources required for channel negotiation for the network based upon the number of secondary users present and the current channel throughput. When the common control channel usage approaches the point at which the additional allotment of resources to rendezvous channel negotiation will create a saturation condition, the network moves to the phase of rendezvous channel negotiation. This method avoids the situation in which common channel saturation is reached, and there are no resources available for additional channel rendezvous negotiation. Therefore, the early channel analysis and start of negotiation prevents the waste of data transmission resources while the common control channel is saturated.

2.5.3 Control Channel Jamming

Control channels facilitate the cooperation among cognitive radio users. As a single point of failure, common control channel jamming (CCC) is the most effective and energy efficient way for an attacker to destroy the entire network system. With common control channel jamming, receivers are prevented from receiving valid control messages when a strong signal is injected into the control channel. This results in denial of service for users of the network [63].

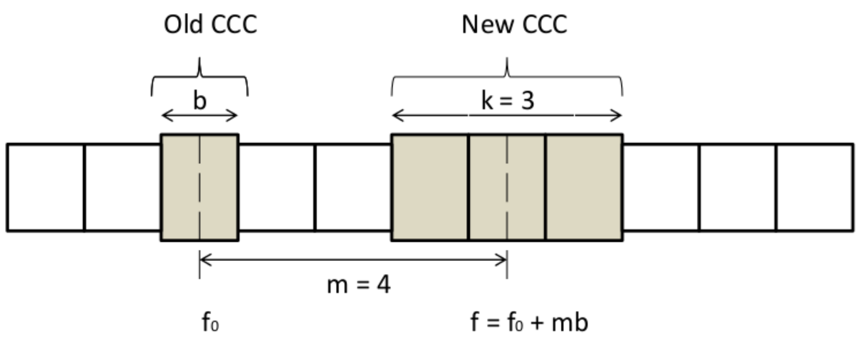

Figure 2.3 Migration of common control channel with (m, k) = (4, 3)

Using dynamic control channel allocation methods combats control channel jamming by maintaining control communications during the attack. There are two methods for dynamic allocation of control channels: cross-channel communication [66] and frequency hopping [54].

The authors of [66] take advantage of the fact that successful communications during a jamming attack can be conducted on another channel not affected by jamming signals. Cognitive radio users can continue to transmit on the channel experiencing interference to notify other network users not experiencing jamming of the new control channel for receiving control messages. This results in successful communication during jamming by using different channels for transmitting and receiving control messages with neighbors. Although communication is maintained, this method incurs high channel switching overhead for radios equipped with a single transceiver.

In the papers [54] and [61] the authors present methods to mitigate common control channel jamming for cluster-based ad-hoc networks using hopping sequences. The cluster-head determines the hopping sequences and identifies the operating control channels for the cluster. Due to the nature of the clustering of the network, the network is partitioned into smaller groups. Therefore, when a jamming attack targets a cluster, the affected network area is reduced. The method presented by [61] differs in that no two nodes share the same hopping sequence.

The mitigation tactic presented by [54] hides the control channel location (frequency), and uses key distribution techniques to allow legitimate users to decrypt the control messages encrypted with keyed hash functions. Control messages are repeatedly transmitted on multiple control channels, so compromised nodes would only have partial keys. Consequently the compromised nodes would be unable to jam all of the control channels. Sufficiently large key distribution with message duplication would therefore allow continuation of control information exchange during jamming attacks.

A polynomial based jamming resilient key assignment protocol is presented by [15]. The key space consists of $p * q$ keys, where p is the number of time slots in a period, and q is the number of control channels. The control transmission is sent repeatedly over all of the control channels in each of the time slots in the period. Each node, including the malicious users, is identified by a unique polynomial. The scheme guarantees access of the nodes to the control channel within a certain time period. However, since the key space must be sufficiently large, based on the number of time slots and control channels, it may incur large control retransmission overhead and delay.

A random key distribution scheme was proposed in [101] and [102] for control channel access under jamming attack. As in [15], the keys are used to hide the control channel allocation in time slots with duplicate transmission on several control channels. The diversity of keys is large, and so it is probable that authorized users hold keys unknown to compromised users. Keys are periodically reused in time slots to limit the key space and corresponding storage overhead. Cryptographic hash functions are used to map the control channel keys to the allocated frequency and time slot for control channel relocation in a reuse period.

The paper [130] provides a method of control channel jamming avoidance without a pre-shared key distribution system. The control data is distributed through cluster heads in the network with each network node belonging to only one cluster. A cognitive radio network with N nodes requires $2log_2N$ keys, with each secondary user receiving log_2N keys based on a unique binary ID. Each cluster head generates and sends two control signals in every time period i. The functions$F(k_i, i)$ and $F(k_i', i)$ are used by the cluster head to determine the control channels, and are known to the cluster nodes. All nodes, including the jammer, receive their assigned keys. Since no two nodes have a full set of matching keys as relates to each time period, the jammer will be unable to prevent any node from transmitting in at least one time period.

Referring to the example from [130] in Table 2.3, assume a malicious jammer, node 3. The jammer can jam the channel determined with k_1' in period 1, the channel determined with k_2

in period 2, and the channel determined with k_3 in period 3. However, since none of the other nodes have the same three keys in the same time periods as node 3, each will be able to transmit on the assigned control channel in at least one of the three periods.

A stochastic general-sum game called jamming-resilient control channel (JRCC) is presented in [64]. The game models the interchange among the cognitive radio users and the attacker under the impact of the primary user. The game objective is to determine the best control channel allocation strategy to combat jamming using multiagent reinforcement learning (MARL). The optimal control channel is found when the game reaches the Nash equilibrium.

In each stage of the game, each radio selects an action that maps to a set of selected common control channels. The nodes receive their rewards by complying with conditions applied to each common control channel. To facilitate cooperation, each radio broadcasts the control message according to the conditions of the channel. Each node's strategies are updated with the parameters received from its neighbors. If the primary user changes the game state, the radios sense the channels to obtain the new state, and update their parameters, learning rate, and strategy. In this manner, the JRCC algorithm enables cooperation between the nodes with low overhead to facilitate common control allocations while adapting to the primary user and learning rates. Simulation results show that the JRCC algorithm effectively combats jamming in an environment that includes primary user activity.

Table 2.3 Key for 8 Secondary Users with 1 Attacker

Node	Unique ID	Key
0	000	$k_1\ k_2\ k_3$
1	100	$k_1\ k_2\ k_3$'
2	010	$k_1\ k_2$'k_3
3	001	$k_1'\ k_2\ k_3$
4	101	$k_1'\ k_2\ k_3'$
5	011	k_1' k_2 'k_3
6	111	k_1' k_2' k_3'
7	110	k_1 k_2' k_3'

Table 2.4 Attacks By Layer - Network

Attacks by Layer	**Network Member?**	**CIA**	**Description**	**Citation**
Network Layer				
HELLO Flood	Internal	A	Node broadcasts HELLO loud enough so all nodes think it is a neighbor. Packets are lost since the node is far away.	[29, 51]
Ripple	External	A	The wrong channel information is provided so that the other nodes in the area change their channel. The attacker's intent is to cause the false information to be passed hop by hop and cause the network to enter a confused state.	[133]
Sinkhole	Internal	C, I, A	Attacker advertises itself as the best route and does selective forwarding in which packets are modified or discarded.	[51, 118, 125]
Sybil	Internal	A	Attacker sends packets as different identities subverting the trust system.	[24, 51, 72, 105, 122]
Wormhole	Internal	C, I, A	Attacker tunnels messages or pieces of messages to different parts of network to replay them.	[44, 51]

2.6 Network Layer

The network layer provides the ability to route data packets from a source node on one network to a destination node on another network, while maintaining quality of service. It also performs fragmentation and reassembly of packets, if required. The cognitive radio network shares security issues with the classic wireless communication networks due to the three shared architectures of mesh, ad hoc, and infrastructure. Cognitive radio networks also share similarities with wireless sensor networks. These include multi-hop routing protocols and power constraints. In addition, there are special challenges faced by cognitive radio networks due to the required transparency of the network activities to the primary user. Routing in the cognitive radio network is further complicated by the requirement of the radio to vacate the frequency when the primary user is sensed as present. Cognitive radio security vulnerabilities are therefore also inherited from these architectural requirements.

2.6.1 Sinkhole

Cognitive radio networks often use multi-hop routing. A sinkhole attacker takes advantage of multi-hop routing by advertising itself as the best route to a specific destination. This activity spurs neighboring nodes to use it for packet forwarding. In addition, the neighbors of the attacker will advertise the offender as the best route, creating a "sphere of influence" for the attacker.

The attacker can begin the attack by building a trust base. The attacker can use a higher level of power so it can send any received packets directly to the base station. It can advertise that it is one hop from the base station, and forward all received packets appropriately for a time. After trust has been established, and advertising of the node as the best route has been propagated through the local area, the perpetrator can begin other types of attacks, such as eavesdropping.

The attacker can perpetrate the selective forwarding attack by forwarding packets from select nodes, or modifying or dropping received packets. This attack is particularly effective with mesh and infrastructure architectures since all local traffic looking to be relayed to another network have the same destination - all traffic leaving the local network needs to go through the base station.

Countermeasures for the sinkhole attack from outside the network are based upon link layer authentication and encryption. Using authentication, an outside attacker will be unable to join the network. Since the cognitive radio network will only use members for routing, the attacker will be unable to advertise as the best route [51].

Countermeasures for the insider attack could be based upon a continually updated trust determination. The cognitive radio network would need a system to monitor dropped or changed packets, and report issues to the fusion center. After analyzing the received data, the base station would flood the network notifying its members of the communication issues recently experienced. It would then drop the attacker as a member of the community.

Additionally, countermeasures to the insider attack can be adopted from wireless sensor network studies, such as the security-aware ad hoc routing protocol (SAR). SAR is based upon on-demand protocols, such as Ad hoc On-Demand Distance Vector (AODV) routing or Dynamic Source Routing (DSR) [125].

With SAR a security metric is added to the route request packet (RREQ) and the route discovery procedure is modified. Intermediate nodes receiving the RREQ packet determine if the security metric or trust level is satisfied. If it is satisfied, the node processes the packet and uses controlled flooding to propagate the packet. If the required security is not satisfied, the packet is dropped. A reply packet (RREP) is generated if an end-to-end path can be found based on the required security attributes. A notification is sent to the sender if such a path cannot be found. The sender can then modify the trust level in order to find a route [125, 118].

With the assumption that a key cannot be determined by nodes that did not receive it from the base, a malicious node that interrupts the flow by altering the security metric cannot cause serious damage. Without the key, the attacker cannot decrypt the packet, and a legitimate node receiving the packet with an altered security level will drop it [118].

2.6.2 Wormhole

The wormhole attack is closely related to the sinkhole attack. Basically, an attacker tunnels messages received in one part of the network over a low latency link. The messages are replayed in another part of the network. In the simplest example, a node situated between two other nodes forwards messages between the two of them. Wormhole attacks are usually administered by two malicious nodes that understate the distance between them by relaying packets along an out-of-bound channel that is unavailable to the other nodes.

A wormhole attack is perpetrated by convincing nodes that are usually multiple hops from the base station that they are only one or two hops away through the adversary. If the end point of the wormhole is relatively far from the base station, most nodes in the local network area will try to use the attacker for forwarding. Packets can then be selectively forwarded to the malicious node close to the base station for additional forwarding, or captured for eavesdropping as they are forwarded [44, 51].

If the adversaries are placed carefully, the attack could result in a partitioned network when the attackers stop relaying the packets. This action would trigger network routing discovery. Participating in the discovery effort may provide the attacker with additional information that could be used for other attacks, such as eavesdropping.

One prevention method for the wormhole attack was suggested by [51]. Karlof and Wagner suggest using geographic routing protocols to forward packets in the network. Such protocols construct a topology based on routing traffic physically towards the base station. Using this routing method, it is difficult to attract traffic towards a sinkhole or wormhole. Local nodes would detect an artificial link because they would notice the distance between themselves and the attacker, or between the attackers, is beyond normal radio range.

The authors of [44] propose using packet leashes to detect and defend against wormhole attacks. The authors present two types of packet leashes: geographic and temporal. Both leashes allow the receiver of a packet to detect if that packet traveled farther than the leash allows.

The geographic leash is used to ensure the packet recipient is within a certain distance from the sender. For the geographical leash to be constructed, each node must be aware of its own location, and the clocks of all nodes must be loosely synchronized. Sending nodes include in their packets their own location and the time the packet was sent. The receiving node compares this data to its own location and the time of receipt. Assuming the clocks of the nodes are loosely synchronized, the receiver can compute an upper bound on the distance between the sender and itself. It is noted that obstacles in the network field would not allow distance bounding based on location data. Therefore, wormholes could still be created, since communication may not be allowed between two nodes that would otherwise be in transmission range.

The temporal leash provides an upper bound on the packet lifetime. This lifetime in effect restricts the maximum travel distance of the packet. Creation of a temporal leash requires tightly synchronized clocks, such that the maximum difference allowed is t. All nodes in the network must be aware of the value of t, and it must be on the order of a few microseconds or less. When sending a packet, the sender would include in the packet the time the packet was sent. The receiving node would compare the time to the time received. From this information, the receiver would be able to determine if the packet had traveled too far based on transmission time and the speed of light.

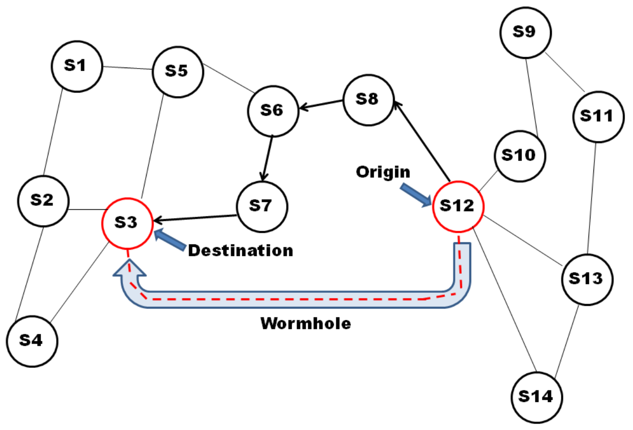

Figure 2.4 Wormhole attack

2.6.3 HELLO Flood

The HELLO attack was first introduced by [51] as an attack against wireless sensor networks. However, due to the possibility of using similar routing strategies, the attack can be applied to the cognitive radio network. The attack is perpetrated by an attacker that broadcasts a message to all nodes in a network. The packet may be advertising a high quality link to a specific destination. Enough power is used to convince each node that the attacking node is their neighbor. The nodes receiving the packets assume the attacker is very close due to the strength of the received signal, when in fact the attacker is a great distance away. Packets sent from the network nodes at the regular signal strength would be lost. In addition, network nodes may find themselves with no neighbors available to forward packets to a particular destination, since all nodes are forwarding packets towards the attacker. Protocols that depend upon localized information exchange between neighbors for topology maintenance are also subject to the attack. Note that an adversary need not to be able to read or construct legitimate traffic - the attacker needs only to capture and re-broadcast overheard packets with enough power to reach every node in the network [51].

The HELLO attack can be defended against by verifying the bi-directionality of links before using the link established by a message received over the same link. Using a base station as a trusted third party to facilitate the establishment of session keys between parties in the network can provide verification of bi-directionality. The session key allows the communicating nodes to verify each other's identity, as well as provides an encrypted link between them. It should be noted the number of shared keys needs to be limited to prevent the attacker from establishing a link between every node. An alarm should be raised about the detection of an attacker if one node claims to be a neighbor to an inordinate number of nodes [29, 51].

2.6.4 Sybil

Local entities that have no direct physical knowledge of remote entities perceive the others as informational abstractions. These are referred to as identities. A system must have the capability to ensure that distinct identities refer to distinct entities [24]. Without this ability, the reputation system used to prevent other types of attacks will be subverted.

An attacker perpetrating the Sybil attack will create a large number of pseudonymous identities so it can gain a disproportionately large influence on the network. In other words, the mapping of identities to entities is many to one. Pairing the Sybil attack with the launch of the primary user and Byzantine attacks can allow the attacker to prevent use of the channel by legitimate users by effectively poisoning the decision making process [104]. Additionally, the misbehavior can be spread amongst the nodes acting as Byzantines, making any one of them especially difficult to identify [72].

Validation of each node's identity is the key to defending against the Sybil attack. The two ways to validate an identity are direct validation, in which a node directly tests whether the identity of another node is valid, and indirect validation, in which nodes that are already verified provide validation or refutation for other nodes.

In [24] resource testing is proposed as a method of direct validation. An assumption made with resource testing is that the resources of the attacker's physical entity are not unlimited.

Identities are tested to verify that each identity has as much of a tested resource as a physical device. The authors proposed measuring the resources available for computation, storage, and communication.

One communication testing example is to broadcast a request for identities, and only accept replies that occur within a given time interval. To test storage resources, each entity is asked to store a large amount of unique, incompressible data. The challenging entity keeps small excerpts of the data to use to verify the challenged identities are storing the data they are sent. Finally, to test computation resources, each entity is asked simultaneously to solve a unique puzzle in a limited time.

The authors of [72] suggest another validating method that may be suitable for cognitive radio networks. For the radio resource testing method, it is assumed each physical device has only one radio. It is also assumed that a radio can only send or receive on one channel at any moment. A node can verify that none of its neighbors are Sybil identities by assigning each of the neighbors a different channel on which to broadcast a message. From the same set of channels, a channel is then randomly chosen by the challenger on which to listen. The challenger will hear the message if the neighbor assigned the channel is legitimate.

In [51] a solution involving symmetric keys is suggested. With this solution, every node shares a unique symmetric key with a trusted base station. The base station also acts as a trusted third party to facilitate the establishment of session keys between parties in the network. The session key allows the communicating nodes to verify each other's identity, as well as establish an encrypted link between them. It should be noted the number of shared keys needs to be limited to prevent the attacker from establishing a link between every node. Also, the base station can place a reasonable limit upon the number of neighbors a node is allowed. An alarm should be raised about the detection of an attacker if one node claims to be a neighbor to an inordinate number of nodes.

As mentioned in [122], many of the trust and reputation based schemes previously proposed can be applied to the Sybil problem. Refer to 2.5.1 for descriptions of these techniques. Nodes with a bad reputation, or those that are proven as untrustworthy, will be punished or removed from the network, regardless of whether they are truly a distinct node, or a Sybil.

2.6.5 Ripple Effect

The ripple effect is a new attack that is specific to cognitive radios because of their ability to change channels during communication. Cognitive radios actively change channels to avoid the primary user and to utilize the channel that will provide the best throughput in the local area. The ripple effect is similar to the primary user emulation or Byzantine attack in that the wrong channel information is provided so that the other nodes in the area change their channel. However, the ripple effect attacker's intent is to cause the false information to be passed hop by hop, and in turn cause the network to enter a confused state.

It should be noted that the attack is especially effective when the attacker transmits with a strong signal because of the following:

1. The activity of a primary user is generally greater than that of a secondary user, so the appearance of a primary user may affect several ongoing transmissions of secondary users.
2. Secondary users expend time and energy for spectrum sensing, neighbor discovery, and channel switching (a few milliseconds) when changing channels.
3. Channel switching of one secondary user may cause a ripple effect, or cascaded switching of multiple secondary users [133].

Countermeasures to the ripple effect attack are similar to those for the primary user emulation and Byzantine attacks. It is essential that primary user presence can be detected and validated. Similarly, it is essential that the information passed from a neighbor about the presence of the primary user is also validated. Such validation can ensure the licensed channel is vacated when necessary, and channel switching will only occur when necessary.

2.7 Transport Layer

The transport layer responsibilities include flow control, congestion control, and end-to-end error recovery. The transport layer in the cognitive radio network is subject to many of the vulnerabilities that plague wireless ad hoc networks.

Table 2.5 Attacks By Layer - Transport and Application

Attacks by Layer	**Network Member?**	**CIA**	**Description**	**Citation**
Transport Layer				
Key Depletion	Internal	C, I	With the great number of session keys created in a cognitive radio network, it is very likely a key will be repeated. Repetitions provide an avenue to break the underlying cipher system.	[67, 86]
Application Layer				
Cognitive Radio Virus	Both	A	The cognitive radio network is vulnerable to viruses that can effect radio function and learning.	[21, 42]
Policy Attacks	External	A	Policy of the radio is changed or not allowed to be updated, providing the attacker unfair spectrum access.	[8]

2.7.1 Key Depletion

CRNs suffer from short transport layer session duration due to high round trip times and frequently occurring retransmissions [86]. This necessarily implies that a large number of sessions are initiated. Most transport layer protocols, such as secure socket layer (SSL) and transport layer security (TLS), establish cryptographic keys at the beginning of each transport layer session. With the great number of session keys generated, it becomes more likely a session key will be repeated. Repetitions of a key can provide an avenue of exploitation to break the underlying cipher system. It has been established that wired equivalent privacy (WEP) and temporal key integrity protocol (TKIP) protocols used for IEEE 802.11 are prone to key repetition attacks.

Security protocols used below the network layer currently are designed to accommodate the total number of sessions that are typically created for wireless LANs. The newer Counter

Cipher mode with block chaining Message authentication code Protocol (CCMP) is designed to exponentially delay key repetitions [67]. CCMP offers enhanced security compared to TKIP by using 128-bit keys with a 48-bit initialization vector. This architecture minimizes the vulnerability of the system to replay attacks. Since the current design is inadequate for the security requirements of cognitive radio networks, new protocols need to be investigated.

2.8 Application Layer

The application layer is the layer closest to the end user. The user and the application layer interact with the application software. The application layer is responsible for determining the resources available, synchronizing communication, and identifying the communicating devices. Cognitive radios require a greater processing power and memory capacity than the traditional smart phone. This is because of the extra tasks performed by the cognitive radio, such as spectrum sensing and learning. Cognitive radios are therefore expected to be the target of software viruses and malware [5]. Additionally, physical and link layer delays due to spectrum handoffs, unnecessary rerouting and stale routing due to network layer attacks and delays due to frequent key exchanges cause degradation of the QoS in the application layer protocols [67].

2.8.1 Cognitive Radio Virus

The cognitive radio network is as vulnerable to viruses as other types of networks and platforms controlled by software. Viruses are computer programs that can replicate themselves and spread from radio to radio. For replication, the virus must be able to execute code and write to memory.

In a self-propagating network like the cognitive radio network a virus can be particularly devastating. A radio infected with the virus can impose upon its neighboring node a false state, or a series of transition states. The neighbor will pass along this false state. A particularly troublesome side effect of this propagation is that an artificial intelligence (AI) cognitive radio will erroneously learn to react to this false environment, affecting future network decisions.

The authors of [42] present a model for the propagation of a self-propagating AI virus through a cognitive radio network. Simulation showed that the time taken to infect the whole cognitive radio network increased exponentially with network size. Second, it was shown that the anti-virus performance of static networks is better than the performance of a dynamic network in the presence of a AI virus. It was also shown that the AI virus propagation speed increases with an available abundant spectrum resource in the area. However, the variability of the spectrum does not affect the propagation speed noticeably.

In the paper [21] the authors suggest a feedback loop into the network to cause the radios to re-learn in the case of propagated false environmental information and consequent decisions and learning. A second approach is to build in logic that will invalidate learned actions that are known to violate certain principles.

2.8.2 Policy Attacks

There are four main functions of the policy system of the policy based cognitive radio. They are policy derivation, policy distribution, policy reasoning, and policy enforcement. The paper [8] describes the security threats associated with each of the functions. The attack on the policy derivation and distribution functions by spoofing, and policy reasoning and enforcement threats, are described below. The policy attacks via forging occur at a different level, targeting the application layer; therefore, those attacks are described under the cross layer attacks.

The functions of policy derivation and policy distribution can be disrupted by a malicious node through spoofing the policy administrator. With the spoofing attack on policy derivation, the faked policy administrator feeds the radio policy manager false or misleading policies designed to decrease network performance or cause interference with the primary user. Similarly, the spoofing attack on the policy distribution function allows a faked policy server to supply misleading policies to the radio's policy engine. An authentication protocol that uses certificates to validate the policy administrator can mitigate these attacks.

The policy reasoning and enforcement attack occurs when a selfish policy controlled cognitive device sends false reasoned information to other ordinary cognitive controlled devices in the area stating there are no available bands for transmission. In this way the selfish device keeps transmission opportunities for itself. Reputation or collaborative decision schemes are recommended as mitigation avenues.

2.9 Cross Layer

Cross-layer attacks launched by adversaries target multiple layers. These types of attacks can affect the whole cognitive cycle of spectrum sensing, spectrum analysis, and spectrum decision. Many of the attacks described earlier can be combined to create cross-layer attacks. In addition, the same attacks may target one layer, but affect the performance at another layer. Often the cross-layer attack will take place on the physical layer while targeting the performance of the MAC layer.

Table 2.6 Attacks By Layer - Cross Layer

Attacks by Layer	**Network Member?**	**CIA**	**Description**	**Citation**
Cross Layer				
Jellyfish	Internal	A	Based on the dual role of the radio as router with forwarding behavior. The attack targets closed-loop flows responsive to network conditions like delay and loss.	[47, 67, 78, 82, 89]
Lion	External	A	Attack utilizes the PUE attack at PHY layer to disrupt the TCP. TCP continues to create logical connections and send packets. The packets timeout, and TCP retransmits. Retransmit timer doubles with backoff resulting in delays and packet loss.	[29, 55]
Routing Information Jamming	Internal	A	A malicious node causes a targeted node to initiate spectrum hand off before the routing information is exchanged.	[67, 134]
Small Backoff Window	Internal	A	Node decreases its own backoff window size so it has a better chance of getting the channel.	[116, 131]

2.9.1 Routing Information Jamming

This attack can take place in a cognitive network with no common control channel. It also takes advantage of the fact that there is delay during spectrum hand off. The delay allows jamming of the routing information among neighboring nodes. The result is the use of stale routes and incorrect routing of packets.

To start the attack, a malicious node causes the targeted node to initiate spectrum hand off before the routing information is exchanged. When spectrum hand off occurs, the targeted node stops all ongoing communication, leaves the frequency, determines a new spectrum for transmission, identifies neighboring nodes, and informs neighboring nodes of the change in frequency. The targeted node cannot receive or transmit updated routing information until the hand off is complete; this is referred to as deafness. Until the routing information is updated, the targeted node and its neighbors will use stale routing information. By causing the targeted node to continuously perform spectrum hand off just before routing information exchange, the attack can be extended and made more severe [67].

The paper [134] presents a collision-free resident channel selection based solution (CF-RCS). A resident channel is selected by each node from the available channel set during network initialization. It then broadcasts this selection with its neighbors. Nodes are expected to receive any updates on the resident channel. However, this protocol requires that each cognitive node is equipped with two half duplex transceivers with one waiting on the resident channel for a request of control message exchange, and the other sitting on the data transmission channel.

2.9.2 Small Backoff Window

The small backoff window attack is also known as the backoff manipulation attack. In this attack the attacker manipulates the contention protocol parameters to retain exclusive or more frequent access to the channel. Selfish or malicious users choose a very small backoff, or contention, window in the effort to gain more access to the channel. This attack is feasible against cognitive radio networks using Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol at the MAC layer.

The authors of [116] base their proposal on the method presented in the first paper, only using a more refined test to compute the difference between distributions. A strategy is presented in which the backoff value of a sender is assigned by the corresponding receiver. Monitoring of the sender's compliance with the assigned backoff window is also provided by the receiver. If the sender deviates from the assigned value, it incurs punishment with the assignment of a larger backoff value for future transmissions. Continued misbehavior can result in the node from being ejected from the network.

The mitigation described above does not apply to events if collusion occurs between the sender and receiver. Neither does it apply if the receiver assigns large backoff values to alleviate contention for its own transmissions. Increasing the number of cognitive radios monitoring the backoff can help alleviate issues of collusion, or the event of the malicious receiver. It was suggested that every cognitive radio publish its backoff schedule in advance, or publish the seed to a publicly known pseudo random number generator used to generate the backoff values. With this information, neighbors can detect misbehavior of neighboring nodes[131].

2.9.3 Lion Attack

The Lion attack is specific to the cognitive radio network. The attack takes place at the physical/link layer, while targeting the transport layer. In essence, the attacker uses a primary user emulation attack in order to disrupt the Transmission Control Protocol (TCP) connection. The attacker can be an outsider or a part of the network.

The attack affects the Transmission Contol Protocol by forcing frequency handoffs in vacating the channel due to the perception that the primary user is present. When the hand off occurs, the TCP is not aware of the switchover. TCP will continue creating logical connections and sending packets while not receiving any acknowledgments. If no acknowledgments are returned, TCP considers the segment as lost due to congestion. As a consequence, TCP retransmits the segment while reducing the congestion window. This results in delays and packet loss, reducing throughput.

The attack can become even more extended and severe, becoming a denial of service attack, if the attacker can anticipate the new channel to which the secondary user will move. If the attacker moves to the new channel, and again simulates the primary user, or jams the channel, the sender will not be able to successfully send data [29].

The authors of the paper [55]present a method of mitigation to the lion attack. Besides identifying the attack, the authors suggest that cross-layer communication must be established in order to make the TCP aware of the attack. This communication will allow the cognitive radio network to halt the TCP connections during frequency hand off. The TCP parameters can then be adapted to the connection parameters after hand off.

Additionally, the control data that is shared by the whole group of cognitive radio network participants needs to be protected from eavesdropping by the attacker to prevent the attacker from becoming aware of the current and future actions of the network. The authors of [55] suggest the use of a common shared secret key. The group key will provide group members the ability to send encrypted data, decrypt received data, and authenticate itself as a network member. Of course, only the current group members should know the group key, so the key would need to be updated as the membership changes. It is suggested current group key management (GKM) studies be applied to the cognitive radio network as a solution. Unfortunately, the cross-layer communication and group key can only mitigate the lion attack since these solutions cannot stop denial of service or channel degradation due to jamming. In an effort to identify the attacker, the authors of [55] suggest adding a parallel cross-layer intrusion detection system adapted to cognitive radio networks.

2.9.4 Jelly Fish Attack

The jelly fish attack and the lion attack are related in that they both target the TCP. In the lion attack, the degradation of the TCP occurs because of frequent frequency handoffs. In the jellyfish attack, throughput is decreased because of out of order, delayed, or dropped packets.

The jellyfish attack is performed at the network layer, while targeting the transport layer. The attacker can perpetrate the attack by intentionally reordering the packets it receives and forwards. TCP has a vulnerability to out of order packets; out of order packets trigger retransmissions and degrade network throughput. Dropping a fraction of the packets also degrades throughput, similar to a sinkhole attack. However, in this variant the packets are dropped intelligently such that they coincide with the TCP transmission window. This can cause near zero throughput in the TCP protocol. Additionally, if the malicious node randomly delays packets, throughput will be affected because it causes the TCP timers to be invalid, resulting in network congestion [67]. Part of the difficulty in mitigating the jellyfish attack is that the jelly fish obeys all of the data plane and control plane protocol rules. Therefore, supportive nodes can hardly distinguish between the attack, and a congested network [78]. It is possible that successful jelly fish attacks can partition the network [89].

In the paper [89] a scheme is presented that exploits the broadcast nature of the wireless medium for detection and mitigation of jelly fish attacks. A jelly fish can be detected by its neighbors simultaneously when the neighbors are set as promiscuous so they can observe each other's activities. In the proposed scheme the TCP protocol is altered such that catalyst-helper packets are sent to check for congestion when the network experiences low throughput. The packets are supplied with cumulative sequence numbers and a flow id number. Observing nodes are able to identify if packets are delayed, dropped, or sent out of order by a neighbor. When a threshold of such detected misbehavior is reached, the misbehaving node is punished, and can be isolated from the network. Punishment can include revocation of the certificate of the malicious node by the centralized trusted authority, or isolation of the malicious node by the dropping of all control and data packets forwarded or originated from the node.

A trust based mechanism is presented in [82] for establishing and managing trust in pure ad-hoc networks where no base station or other central entity exists and the nodes are not required to be pre-configured. Routing protocols, such as Dynamic State Routing (DSR) and Ad-hoc On-demand Distance Vector (AODV), are modified to allow establishment of routes with a certain level of confidence. Nodes first check the trust value at the next hop to ensure it is equal to or greater than a specified threshold before forwarding packets to the node. If the threshold value is not adequate, the sending node will try to avoid a path using the suspect node [82].

A scalable and a robust approach to enforce collaboration in a mobile ad hoc network is presented by [47]. In this mitigation effort, every node observes its neighbors' activities. Each node computes the ratio of dropped packets in a certain time window for its neighbors that drop packets. When a ratio for a node exceeds an predetermined threshold value, the one-hop neighbors punish the node with isolation for a time period.

2.9.5 Policy Attacks

As mentioned in 2.8.2, the paper [8] describes the security threats associated with each of the main functions of the policy system of the policy based cognitive radio. The attack on the policy reasoning and enforcement functions, as well as the policy derivation and distribution functions by spoofing, were already described. The forging policy attacks occur at a different level, targeting the application layer, and are described here.

In the forgery attack against the policy derivation function, the malicious entity intercepts communications from the policy administrator intended for the policy manager. The original policy is replaced with a forged policy resulting in a compromised network and decreased network performance. Similarly, the forgery attack on the policy distribution function intercepts and replaces the policy from the policy server intended for the policy engine. The use of certificates or other authentication protocols for identity validation can mitigate these attacks.

2.9.6 A Suggested Multi-Level Security Framework as Attack Mitigation

Trying to address several layers of attack of the cognitive radio network, the paper [91] presents a multi-level framework for the security of the cognitive radio network. The basis of the proposal is a new, secure, adaptive MAC protocol called dynamic decentralized and hybrid MAC (DDH-MAC). This cognitive radio MAC protocol is a hybrid that lies between the static common control channel using an unlicensed spectrum band (commonly referred to as GCCC) and the non-GCCC protocols. The protocol creates an adaptive, secure, and energy efficient network by tuning its parameters efficiently and intelligently based on the current situation of the network. The protocol includes a primary control channel and a backup control channel, both sent over the white spaces in the spectrum.

Four levels of security are provided by the DDH-MAC protocol. First is the encryption of the beacon frame. Recipients of the beacon frame apply the relevant decryption scheme to read the primary and backup control channels.

The second level of security is the secure transmission of the free channel list (FCL). The FCL is exchanged secretly over the primary control channel. The chosen control channel is only known to the cognitive radios in the vicinity. Additionally, all frames are encrypted using the public key, and only nodes with the private key can retrieve the information.

Dynamic decentrlized and hybrid MAC adds a time stamp to each data transmission as a third level of security. Data is expected to be received in a certain period of time; if the data is not received in the specified time period, it is assumed the integrity of the data could be compromised and therefore untrustworthy. This protocol helps protect the system against man-in-the-middle attacks.

The last level of security is the dynamicity of the control channel. Since the primary control channel is sent over a white space, the appearance of the primary user could occur, thus moving the network communication to the backup control channel. If the primary user also appears on the backup control channel, the nodes switch to the GCCC to search for a beacon frame. Any attacker targeting the primary and backup control channels via smart jamming will need to re-compile their attack strategy whenever the primary users appear. This provides a higher level of security to the network.

2.10 Conclusion

With our increasing usage of the air as a medium for connecting electronically with the world, the current spectrum defined for commercial and personal usage has become crowded. The cognitive radio network with software defined capabilities will open to users more spectrum frequencies, and hence, enhanced communication opportunities. However, the new technology also provides avenues for new attacks perpetrated by malicious or selfish users with the desire to inhibit communication, capture or change the message, or use the spectrum exclusively.

In this paper we have presented the structures of malicious attacks on the cognitive radio network. We have identified attacks from both the traditional cellular networks and the wireless sensor network arena that apply. We also presented attack scenarios specific to the cognitive radio network architecture and capabilities. Following each attack scenario we presented mitigating techniques particular to the attack.

Recent security research on the cognitive radio network has focused on the insider threat (Byzantine), jamming of the control channel or other portions of the spectrum, and externally affecting spectrum usage by masquerading as a primary user. More research needs to be completed in the area of secure transport protocols for the spectrum-aware cognitive radio networks, considering the network's unique characteristics in spectrum management and spectrum mobility. Additionally, research needs to take place in the realm of cognitive radio ad hoc networks (CRAHNS), addressing their distinctive security issues related to their network building functions. Finally, further research needs to be conducted in the area of protecting the cognitive radio function from many of the traditional threats, such as worms, Trojans, and viruses, as well as new threats that attack the radio's ability to learn.

As the cognitive radio network concept matures and comes to fruition, the network security sword play of thrust and parry will continue. The true challenge of the security warrior is prior preparation for the battle. Extensive research and discussion about securing the network will contribute to a proper framework that can be built into the cognitive radio system.

CHAPTER 3. DESIGN AND ANALYSIS OF A METHOD FOR SYNOPTIC LEVEL NETWORK INTRUSION DETECTION

A paper accepted by *COMPSAC 2015: The 39th Annual International Computers, Software & Applications Conference*

Deanna T. Hlavacek[1][2][3] and J. Morris Chang[4]

3.1 Abstract

Current system administrators are missing intrusion alerts hidden by large numbers of false positives. We propose an intrusion detection tool that effectively uses select data to provide a picture of "network health". Our hypothesis is that by utilizing the data available at the node and network levels we can create a synoptic picture of the network providing indications of many intrusions or other network issues. Our major contribution is to provide a revolutionary way to analyze node and network data for patterns, dependence, and effects that indicate network issues. Our first contribution in this vein is to present a method based on utilizing the number of packets sent, number of packets received, node reliability, route reliability, and entropy to develop a synoptic picture of the network health in the presence of a sinkhole.

3.2 Introduction

Wireless ad hoc networks are self-organizing and self-configuring infrastructure-less networks of nodes which are connected by wireless links such as 802.11/WiFi products, wireless sensor networks (WSNs), and the new cognitive radio network. With the growth in popularity of these

[1]Graduate Student, Department of Electrical and Computer Engineering, Iowa State University.
[2]Primary Researcher and Author.
[3]Author for correspondence.
[4]Associate Professor Department of Electrical and Computer Engineering, Iowa State University.

technologies there is a growing demand for intrusion detection systems (IDS) that can operate with network node cooperation. Current research on intrusion detection systems for wireless ad hoc systems focus mainly on anomaly or signature detection. These methods are subject to high false positive rates. With limited time and resources, many true positives are lost in the overload of the combined true and false alerts. Perhaps the answer is not more data, but the better use of existing data.

Our motivation for research related to intrusion detection arises from the current lack of comprehensive research into methods of analysis of selective information in an effort to construct a big picture of network security and integrity, termed as "network health". Our major contribution is to provide a revolutionary way to analyze node and network data for patterns, dependence, and effects that indicate network issues. Our first contribution in this vein is to present a method based on utilizing packet delivery ratio (PDR), node reliability, route reliability, and entropy to develop a synoptic picture of the network health in the presence of a simple sinkhole. In order to provide a proof of concept we utilize a simple grid based stationary network similar to a wireless sensor network. With this simplified first example we intend to show that, although the concept of intrusion detection is not revolutionary, the method in which we analyze the data for clues about network intrusion and performance is innovative, and can be a valuable addition to the intrusion detection "toolbox".

In this paper we will first take a look at current research related to intrusion detection systems designed for wireless ad hoc networks. We then provide a description of our methodology in network analysis for sinkhole detection, and our results based on simulation data. Last we will conclude with a synopsis of the process and the impact of our experimental results. The sections of the paper are organized as follows: Section 3.3.2 describes current research into intrusion detection systems and sinkhole detection in wireless ad hoc networks; Section 3.4 presents the methodology of identifying a sinkhole based on the PDR, node reliability, and system entropy; Section 3.5 presents our results with simulation and data analysis; Section 3.6 provides comparison of the simulation results to other sinkhole identification methods; and Section 3.7 is the conclusion.

3.3 Related Work in Wireless Ad Hoc Networks

Wireless ad hoc networks have no router or access point providing infrastructure to the network. Each node provides routing services, via routing protocols, by forwarding packets to their neighbors. All nodes in an ad hoc network have equal status in the network, and can associate with any network device within range. There are three main routing protocols for ad hoc networks: Optimized Link State Routing (OLSR), Dynamic Source Routing (DSR), and Ad Hoc On Demand Routing (AODV). OLSR is a proactive, or table-driven, protocol. DSR and AODV are both on-demand, reactive protocols in which the nodes maintain routing tables. AODV's routing tables are refreshed according to a timer. We have chosen to use the AODV routing scheme for our first demonstration.

3.3.1 Intrusion Detection Systems

Conventional intrusion detection systems (IDS) are based on misuse detection, anomaly detection, or deviation from specifications. Misbehavior/misuse detection refers to identifying an attack by comparing collected information against a predefined list of "signatures" of known attacks. Anomaly detection is a close opposite to misuse detection. With anomaly detection, rather than storing a list of the signatures of known attacks, the system stores patterns of "normal" behavior for comparison to the current behavior. The third technique, specification comparison, also compares the current behavior to a stored behavior profile. However, the comparison is against manually defined specifications, rather than machine learning techniques.

Recently, most of the intrusion detection system research for wireless ad hoc systems has focused upon the detection of anomalous behavior patterns. In the paper [110] a new detection scheme called AODVSTAT is presented. The method is similar to other watchdog schemes in that the nodes watch the packet events and the meta-data in the packets for anomalies in the protocol. Deviations from the protocol are considered state changes, and trigger an alarming event. In an effort to lower false positive rates using packet features as the basis for anomaly detection, the authors of [58] conduct careful feature selection from the available set of packet features for their method. Similarly, [53] uses the entropy of packet features to detect deviations, and provides a ranking of alerts in an attempt to lower the false positive rate. The authors of [108] base their system solely upon the packet sequence number mismatch with the expected packet sequence number. A confidence level for each node is calculated based on the number of interactions with its neighbors. Rather than packet features, the authors of [135] base their detection on anomalies in traffic patterns, comparing the current traffic pattern to a learned traffic pattern. Each of these techniques requires either knowledge stored in memory of the "normal" pattern of behavior, or some type of training before deployment. Attacks that do not register against the learned normal profile can not be detected by these systems. These methods are also prone to false positives, since hiccups in the network can cause the systems to react with identification of pattern anomaly.

The authors of the papers [3, 34] both present hybrid detection schemes. The paper [3] combines anomaly and misuse detection schemes in order to lower the false positive rate generally seen with anomaly detection, and raise the low detection rate ascribed to misuse detection. The authors of [34] chose to combine the anomaly and specification based schemes, and uses a reputation based system in an attempt to lower the false positive rate. However, this method adds much packet overhead as the nodes in the neighborhoods vote.

In comparison to these methods of intrusion detection, our intrusion detection method is based not on stored patterns, signatures, or rules, but the effect upon the node, route, and network function. Our system requires no training or pre-placed data concerning the expected network protocol or traffic pattern. Therefore, new or craftily tweaked older intrusion methods, such as a stealthy sinkhole that follows network protocols while selectively dropping packets, will still be identified, as long as the effect of the disruption surpasses the established threshold. We have no watchdogs observing the network, and so we are not plagued by the high false positive rates endemic to these methods. Similarly, we are not doing signature comparison, so we are not plagued by the inability (and associated low accuracy rate) to recognize new attack signatures. Our system instead is monitoring the state of the network, and reacts when the state of the whole, or portions of, the network move out of alignment. As in holistic medicine, we do not only pay attention to the symptom; we analyze the symptoms to explore the network function and find the root cause of the malfunction.

3.3.2 Sinkhole Detection

We consider sinkholes as Byzantines in the network. Byzantine behavior is displayed by any action of a member node that negatively affects the routing service in the network. Many such attacks, such as eavesdropping or packet modification, can be prevented by traditional authentication, integrity, and encryption mechanisms. The malicious actions of a Byzantine sinkhole may be more complex, such as modifying the hop count, sequence number, or list of nodes in a path, in order to make itself more attractive as an entry to an ideal route. According to the paper [6], attacks using these tactics can also be prevented with more sophisticated authentication and integrity techniques. We therefore consider the stealthy sinkhole that drops data packets, entirely or selectively, while participating in the routing protocol.

The authors of [6] present a method of identifying a sinkhole with link weights and probes. However, this method is part of a newly proposed routing protocol that includes double flooding during route discovery and the sending of probes to all network nodes for attack discovery. These steps create additional network overhead. Additionally, in this routing protocol, the sinkhole will only be discovered if it is acting maliciously during the probing phase. Finally, this work provides no insight as to how to identify a sinkhole in the accepted ad hoc routing protocols Ad Hoc On Demand Distance Vector Routing (AODV), Dynamic Source Routing (DSR), or Optimized Link State Routing (OLSR).

In the paper [57] the authors present a packet drop attack detection method in which the neighbors adjacent to a communications route monitor the actions of the en route nodes. If a particular node does not forward a specific number of packets in a certain time period, an alert proclaiming a malicious node is started. The authors make a distinction between greyholes, which drop only a portion of the packets, and blackholes, which drop all received packets. Analysis by the authors indicates that only blackholes, and gray holes in the same vicinity, can be identified. If no blackhole exists, the greyholes will not be identified. Similar to this method is the watchdog method presented by the authors of [39]. Once again, collaborative nodes observe the actions of the nodes en route and use a protocol to determine when an alarm should be sounded. Unfortunately the reliability and effectiveness of this method is difficult to determine since the authors have not yet determined the false positive/false negative rates for the protocol.

Several papers rely upon the sinkhole bucking the routing protocol by changing the sequence numbers. The sinkhole identification schemes described in the papers [22, 33, 46] are therefore not effective in identifying a stealthy sinkhole that does not change the packet sequence number. The authors of [22] additionally use the previous image ratio to identify the sinkhole. In the previous image ratio method the received routing packets are compared to other stored routing packet images. However, this method relies upon the sinkhole having forged the route records in their route request packets. Therefore, if a stealthy sinkhole has not forged the route records, the sinkhole will not be identified by this method.

A third indicator of a sinkhole identified and used by the authors of [22, 46] is the route add ratio. The route add ratio is the number of routes that traverse a particular node divided by the total number of routes added to the node's routing table. Unfortunately, [46] only mentions the idea of the route add ratio, but does not explain how the ratio is used. In [22] a network node is specified to keep a counter for each node in the network, and increment the counter when a route passing through the node is added to the cache. This presents the issue of one node storing data for all of the nodes, and additional messages created and sent to the assigned node when any network node adds a route to its cache. The data related to this study did not provide a method to determine the message overhead related to this technique.

In comparison to these studies, our method does not rely solely upon the sinkhole cheating on the routing protocol. Therefore, a stealthy node will still be identified by its effect on the network, rather than missed due to it not sending signals through the routing protocol. Also, even though there are additional messages included with our sinkhole identification scheme, no additional messages are required before the possibility of a sinkhole is discovered. The overhead of messages m related to sinkhole identification is related to the number of hops from the detecting node to the sinkhole and is very small. We analyze the number of messages required in Section 3.6.3.

3.4 Methodology

The methodology of the sinkhole identification scheme consists of two parts. First, the detection phase, in which one or more nodes are alerted that there is a possible sinkhole in the network. Individual nodes calculate their neighbors' reliability values, and are alerted when a reliability crosses a threshold. The second phase is the sinkhole identification phase and involves querying specific network nodes for data they have about *their* neighbors. We make the assumption that each node is aware of its immediate neighbors. However, the nodes may not immediately be aware of the position of its neighbors relative to itself or each other.

Algorithm 1 Calculations of Neighbor Reliability

```
//Done for each neighbor node (Y) around the starting node A
//routeRel: route reliability
//neighRel: neighbor reliability
1 : for each neighNode(Y)
//Calculate route reliability for routes X through neighNode Y
2 :   for each route(X) (Z nodes along route)
3 :     for each node(Z) on route X
4 :       routeRel(X) = routeRel(X) * pdr(Z);
5 :       saveRouteRel(XZ) = routeRel(X);
6 :       get next node pdr(Z) on route(X);
7 :    get next route(X);
//Calculate neighbor reliability using all routeRel(X)
8 :   for each routeRel(X)
9 :     neighRel(Y) = neighRel(Y) +
                    (routeRel(X) * [$\log_2$(1/routeRel(X))]);
10:     saveNeighRel(X) = neighRel(Y);
11:     get next routeRel(X);
12: get next neighNode(Y);
```

3.4.1 Detection Process

In the neighbor reliability method of determining the health of the network, each node counts the number of packets sent and received along each route. The nodes calculate the packet delivery ratio for the stored routes by relating the number of packets received along the route to the number of packets sent along the route. This PDR is also referred to as the "route reliability", and represents the cumulative reliability for each node along a route. From this data, the nodes each determine their "neighbor reliability" by calculating the entropy for all known routes through the neighbor node. Neighbor reliability refers to a single node's perception of the probability of a packet following any route through a single neighbor to successfully reach the intended destination. The Shannon entropy equation, used to estimate the diversity of the system, is applied. The formula follows, where p(x) is the route reliability:

$$H(x) = -\sum p(x) log_2 p(x) = \sum p(x) log_2(1/p(x)) \tag{3.1}$$

Algorithm 1 describes the process to obtain the neighbor reliability values. Note that in simulation we use the probability that a packet will reach its destination when routed through an individual node for the PDR of node Z. This was substituted for the true PDR of a route that would be known in a live network.

Using this method, assuming a stationary MANET grid of nodes with one node per grid space, each node in the grid will have up to eight different neighbor reliability values, each value from the perspective of one of its (up to eight) neighbors. Additionally, each node will have up to eight neighbors for which it has determined neighbor reliability values, comprising the neighbor reliability set. Using its neighbor reliability set, each node calculates the standard deviation of the set. A lower boundary is calculated by subtracting the standard deviation from the mean of the set; the boundary acts as a threshold. The use of the standard deviation was determined experimentally; one standard deviation provided a proper boundary to determine if the neighbor reliability was low enough to indicate the possibility of a sinkhole when compared to the neighbor set. The lower boundary in a live network will need to be determined based upon the particular network. This process provides the node the ability to observe the current state of the network surrounding it, and helps identify anomalies based on the current network state. If the neighbor reliability of any neighbor crosses the threshold, the node is alerted that there may be an attacker in the network. Note that until this point, all calculations and decisions are made upon data collected by the node without additional messages or queries to the network or neighbors. Therefore, unless an alert is signaled, there is no impact upon the network throughput for this method.

The example in Figure 3.1 shows several nodes with reliability values applied to each node *by its neighbors*. The values in the figure are charted according to compass coordinates from the perspective of an individual node. For example, Node W applies the values of 5.443 to node

AA, 5.324 to node BB, 5.436 to node CC, 5.276 to node X, 5.443 to node V, 5.393 to node R, 5.441 to node Q, and 5.329 to node S. These values comprise node W's neighbor reliability set. Conversely, node W is applied the values of 0.1322 as perceived by node AA, 0.1315 by node BB, 0.1331 by node C, 0.1333 by node X, et cetera.

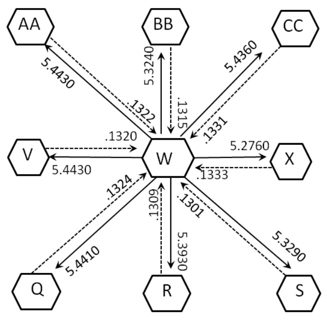

Figure 3.1 Neighbor Reliability Plot (values * 10e-6)

3.4.2 A Proof

We consider a simple grid network of nodes with a sinkhole in the center (Figure 3.2). Let a be the percentage of successful packet delivery for regular nodes; it is assumed regular nodes drop few packets. The sinkhole drops a large number of received packets; let the percentage of successful packet delivery for the sinkhole be b. The network uses AODV routing, utilizing the shortest route. We make the following assumptions:

1. Each node is aware of all of its eight neighbors.
2. No route will pass through more than one of the source node's neighbors, and no more than two neighbors of any node along the route.
3. All routes are of three hops.
4. Percentage $a >> b$.

A source node (src1) located next to the sinkhole will have seven neighbors with routes that do not traverse the sinkhole. The source node will therefore experience a packet delivery percentage of a. The last neighbor will have all packets routed through the sinkhole, experiencing a packet delivery percentage of b. The average percentage of packets experienced by this node will therefore be:

$$PDR_{src1} = (7a + b)/8. \tag{3.2}$$

A source node (src2) two hops from the sinkhole will have more routing choices that do not include the sinkhole. In the described network, five of the neighbors will have no routes through the sinkhole. Of the three neighbors left, two will each have four of their nineteen routes traversing the sinkhole (see Figure 3.2). Let this be represented by $PDR_{31/33}$ since the description applies to both of these neighbors in the example. One neighbor will have three of its thirteen routes passing through the sinkhole (Figure 3.3). Let this be represented by PDR_{32}. The PDR experienced by src2 is therefore represented by the following:

$$PDR_{src2} = 5a + PDR_{31/33} + PDR_{32} \tag{3.3}$$

where

$$PDR_{31/33} = [2(15/19)a + 2(4/19)b]/8 \tag{3.4}$$

and

$$PDR_{32} = [(10/13)a + (3/13)b]/8 \tag{3.5}$$

or

$$PDR_{src2} = [7.53a + .65b]/8. \tag{3.6}$$

Likewise, a source node (src3) three hops from the sinkhole will have more routing choices that do not include the sinkhole. In this scenario seven of the neighbors do not have any routes through the sinkhole. One neighbor has three of its thirteen routes traversing the sinkhole. We get the following equation:

$$PDR_{src3} = [7a + (10/13)a + (3/13)b]/8 \tag{3.7}$$

which is equivalent to:

$$PDR_{src3} = [7.77a + .23b]/8. \tag{3.8}$$

Since $a >> b$, we remove the terms with b from the equations, resulting in the relationship $PDR_{src1} < PDR_{src2} < PDR_{src3}$ since

$$7a < 7.35a < 7.77a \tag{3.9}$$

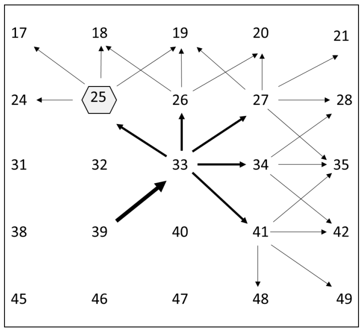

Figure 3.2 Route Example from SRC2 (Node 33)

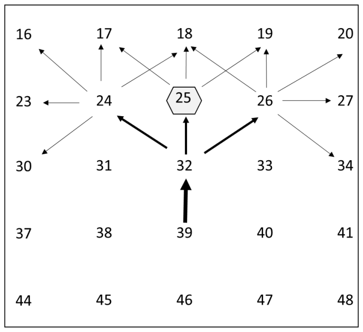

Figure 3.3 Route Example from SRC2 (Node 32)

3.4.3 Sinkhole Identification Process

The sinkhole identification process works similar to water in a reservoir. When the water reaches a higher point in the reservoir, it stops moving forward and splashes back. In our identification process, the query messages will move towards the lowest reliability point and stop at the higher value nodes on the other side of the lowest point. Once the higher nodes are reached, the "splash" will be a broadcast message flooded through the network reporting the identity of the sinkhole.

In the identification process, we assume that each node is aware of its immediate neighbors, although not necessarily their relative positions. We call this set of neighbors "ring one" as related to the alerted node. When a node determines that a threshold has been crossed, it broadcasts a query to its immediate neighbors (ring one) for their neighbor lists. The query includes the alerted node's assignment of L_1, which is the node with the lowest neighbor reliability from the perspective of the alerted node. Only the neighbors with neighbor lists that include L_1 broadcast a reply to the query. Node L_1 was able to receive the information sent by its own immediate neighbors in response to the query made by the alerted node. Node L_1 assigns L_2 to its neighbor with the lowest reliability value (L_2 will most likely, but not necessarily, be in ring two), and sends this assignment along with a broadcast query for neighbor lists. Only the neighbors with L_2 in their neighbor lists that have not already broadcast their neighbor lists during this identification process round reply to the query. (Algorithm 2)

This process continues until we have reached the area around the sinkhole, ring n. The nodes queried in ring n will include the sinkhole and some of its immediate neighbors. At this point the sinkhole may or may not participate in the process. If it does participate, the sinkhole will continue the process by identifying L_{n-1}. L_{n-1} will then identify the sinkhole as its neighbor with the lowest reliability value. The sinkhole may not acknowledge, nor realize, it is the sinkhole. Therefore, the sinkhole (call the sinkhole S) assigns L_{LOW} (call this node X) to one of its neighbors. Node X compares its neighbor reliability values and re-identifies the sinkhole (S) as its neighbor with the lowest neighbor reliability. To attain confirmation, X broadcasts a query to its neighbors for a "vote" as to which of the two nodes they identify as the node with the lowest value. Only nodes with both nodes S and X as neighbors reply. The node requesting the vote will flood the network with the identification message naming the sinkhole. If the sinkhole does not participate, the node (L_{n-1}) that sent the message to the sinkhole and its neighbors will note that there have been no messages from L_n in time T. At this point, L_{n-1} will resend a message to the sinkhole and neighbors common to the requesting node and the

Algorithm 2 Identification of Sinkhole

```
1 : initialize row, col, x to 0; AlertedNode is L0 at [0,0]
3 : Lx identifies Lx+1 as neighbor with the lowest neighbor value
4 : Lx broadcasts query for neighbor lists along with Lx+1 identification
5 : x=x+1
// if Lx participates in the discovery process it will automatically query neighbors
6 : while sinkhole not found
7 :   Lx identifies Lx+1 as neighbor with the lowest neighbor value
8 :   if Lx+1 == Lx-1 OR Lx+1 == Lx-2 AND if other shared neighbors that have not yet been polled exist
9 :     immediate neighbors (not Lx-2, Lx+1, or Lx-1) vote to determine which of the compared nodes has the lowest value
10:     node that is not determined lowest value alerts network of identity of sinkhole
11:   else if y > 2 AND Lx+1 == Lx-y
12:     Lx-1 removes Lx neighbor value from neighbor list
13:     x = x-1 //this is a loop with no positive sinkhole determination
14:   else if Lx queries neighbors for THEIR neighbor list
15:       Lx+1 receives neighbor lists from immediate neighbors and determines relative placement of immediate neighbors
16:  else if Lx-1 does not hear Lx query neighbors in time T
17:     Lx-1 queries immediate neighbors shared with Lx (Lshared_1 and Lshared_2) for their neighbor lists and neighbor values
18:     Lx-1 determines relative locations of neighbors shared between Lshared_1 and Lshared_2
19:     if neighbors shared include only Lx and Lx-1
20:       Lshared_1 and Lshared_2 send queries for neighbor lists and reliabilities
21:       neighbors with lists that include Lx respond to query
22:       Lshared_1 and Lshared_2 provide query information to Lx-1
23:     Lx-1 uses neighbor locations and neighbor values to determine if Lx has lowest local neighbor value
24:     if Lx has lowest neighbor value as reported by surrounding immediate neighbors
25:       Lx-1 alerts network Lx is a sinkhole
26:     else
27:       Lx-1 removes Lx neighbor value from neighbor list
28:       x = x-1
29:   x=x+1
30: end while
```

non-responding (possible) sinkhole, requesting the reliability values for all of their neighbors. If the neighbor lists received do not include at least three neighbors in common, additional queries will be sent by the neighbors of L_n to their neighbors. The additional information received will be provided to L_{n-1}. Analyzed together, this information will confirm that the node with the lowest reliability in ring n is indeed the sinkhole, and L_{n-1} will broadcast the message identifying the sinkhole to the network.

The action taken by the network upon identification of the attacking node is dependent upon the system protocol, and is beyond the scope of this paper. However, there are basically two types of actions: isolation of the node, or incentivation for proper network behavior. Isolation means the suspect node may be ignored by the network when it advertises access to routes to any destination, and so ostracized from the network. Incentivation means allowing the suspect node to send its own data only at the rate that it is providing to the network, and this allowed rate increases as the misbehaving node decreases the number of dropped packets and increases its packet delivery ratio. It must be recalled that although we have referred to the suspect node with low reliability an attacker, it may be a selfish node, or just a malfunctioning node. This method of identifying the unreliable node does not determine intent.

Table 3.1 Network Parameters

Node Parameters	Value
Number of Nodes	30
Node Placement	6 by 5 grid
Simulation Duration	1500 seconds
Routing Protocol	AODV
Type of Stations	MANET
Node Speed	0 kts
Transmit Power	.0001 w
Packet Reception Power Threshold	-95 dBm
Buffer Size (Member Nodes)	256000
Buffer Size (sinkhole)	415
Route Length	5 nodes, 4 hops

3.5 Simulation

To obtain simulated network data we chose OPNET Modeler 17.5, a commercially available tool set used by the communications industry for modeling, simulation, and analysis of communications networks and applications. The initial network topology is a six by five grid mobile ad-hoc network (MANET). The size of our network was chosen based on a study of the effects of insider attacks by the authors of [28]. According to the study, thirty nodes is the ideal size of network for a sinkhole to operate effectively.

In the initial study the nodes are stationary and the parameters for the member nodes are uniform. Traffic supplied by the OPNET MANET model is used. Each node is a traffic generator. No specialized traffic is added, and no additional noise is added to the network.

Ad hoc on demand (AODV) routing is chosen as the routing protocol. Route length for the identification method is limited to five nodes, with four hops. This limitation was imposed to provide a simple initial standard for comparing route reliability results. The routes used for the calculations are not all inclusive. Therefore, not all neighbors may play a part in determining the calculated neighbor reliability value. Parameters for the member nodes are shown in Table 3.1.

The buffer size for the member nodes was left at the OPNET MANET default of 256000 packets. The buffer size for the attacker was lowered to 415 packets to simulate a sinkhole. This buffer size was chosen because it allowed packets to be dropped without stopping all traffic routing through the attacking node. The number of dropped packets at this setting impacted significantly the amount of data traffic sent by the attacking node. Normal nodes sent an average of 903.5 packets over the 1500 second simulation period, as opposed to the 17.3 packets sent by the sinkhole. Note that the 1500 second simulation period and the 15 second intervals were arbitrary. Since the sinkhole effect is recorded almost immediately and the effect is nearly constant over the entire period using the simulation parameters described, the interval parameters for recording the data can be optimized. Subsequent tests showed that the sinkhole could be detected in the first fifteen seconds of the commencement of network traffic. Data collected per node included:

1. At each node, the number of packets received per second in fifteen second intervals.

2. At each node, the number of packets sent per second in fifteen second intervals.

3.5.1 Discovery and Identification Processes Using Simulation Results

In this context, the packet sent ratio (PSR) is the ratio of the number of data packets sent to the number of data packets received at a particular node. To find the neighbor reliability in our simulation, we first calculate the route reliability by multiplying the PSRs for each node along a route, starting at the first hop (as opposed to the sending node). Next we calculate the entropy of all known routes through the neighbor node. Plotting the neighbor reliability values on a plot of the network shows that the lower values tend to be centered around the sinkhole.

Table 3.2 Node G's Neighbor Reliability

Starting Node: G	Cumulative Reliability: 0.000258769			Average Reliability: 4.31282E-05		
Reliability Neighbor List	A	B	C	H	L	M
Reliability Value List	3.05418E-06	2.95563E-06	3.05779E-06	2.21184E-06	2.62348e-06	2.2152E-06
Entropy Per Neighbor List	5.59549E-05	5.42868E-05	5.60144E-05	4.07334E-05	4.82379e-05	4.07903E-05

For an example, we assume node G is the first node to identify a possible attacker. From the reliability calculations (Table 3.2), node G identifies neighbors H and M as having entropy values below the threshold value for node G. Node G currently has a view of the network that includes its immediate neighbors, although node G may not know the relative locations of the neighbor nodes at this time. Node G identifies node H as the neighbor with the lowest reliability value. Node G broadcasts this assignment and queries its neighbors for their neighbor lists.

Only G's neighbors that share H as a neighbor broadcast their neighbor list; therefore, nodes L, M, C, and B broadcast their lists. Upon receipt of the data, node H places the nodes on the grid relative to itself (3.4). Node H reports it's neighbor with the lowest neighbor reliability value to node M, and queries its own neighbors for their neighbor lists. Again, only the neighbors that share node M as a neighbor broadcast their neighbor lists. From this information, node M is able to place itself and its immediate neighbors on the grid. Node R is assigned as having the lowest neighbor reliability value by node M. Neighbors that share node R as a neighbor make their neighbor list reports. Node R identifies node W as the neighbor with the lowest reliability value, continues the process, and node W identifies node X. Node X then identifies node W as the neighbor with the lowest reliability. Since nodes W and X identified each other, shared neighbor nodes "vote" for the neighbor with the lowest value. Since W has participated in the process, and it has been identified as the neighbor with the lowest reliability in the local area, node W (or, if needed, node X) announces to the network W is the sinkhole.

Table 3.3 Partial Table of Neighbor Reliability Values (*10e-5

Node H Neighbors	L	N	M
Reliability Values	4.73898	5.32743	3.51034
Node M Neighbors	Q	R	S
Reliability Values	5.56492	1.90244	5.39725
Node R Neighbors	V	W	X
Reliability Values	5.29084	.130932	5.31013
Node W Neighbors	X	BB	CC
Reliability Values	5.27565	5.32431	5.43592
Node X Neighbors	W	BB	DD
Reliability Values	.133259	5.23300	5.29058

We borrow a technique from weather forecasting to analyze the plot by drawing isopleths for the reader to indicate the levels of entropy in the network. This provides a human-readable synoptic view of the network at the current time. As can be noted in Figure 3.4, the sinkhole has been identified as the lowest area of reliability in the network. Even though the immediate neighbors to the sinkhole node are not dropping packets, their proximity to the sinkhole affects their reliability value because of the number of routes they have stored for use in their routing table that traverse the sinkhole. Nodes farther away from the sinkhole have their reliability values less affected by the attacker because they tend to have fewer routes traversing the sinkhole.

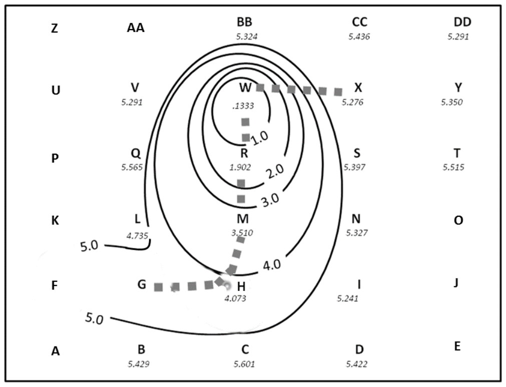

Figure 3.4 Synoptic analysis of neighbor reliability

3.5.2 Data Analysis - One Sinkhole

Analysis of the data set shows that every node except node E resulted in an alert due to at least one of the neighbor nodes having entropy less than the one sigma lower boundary threshold. It must be noted that node E did not have any routes traversing node W (the sinkhole) in its routing table. Every other node had at least one route traversing node W. Also, starting the identification process from every node (except E) found W to be the sinkhole.

There are cases in which the identification process may not be initiated and the sinkhole not found. However, as the network matures and nodes store routes with the sinkhole traversed, the identification process will be initiated. The special cases are listed below:

1. If a node exists with no routes stored that traverse the sinkhole, the sinkhole will not be detected by this particular node.

2. If a node exists with all routes traversing only one neighbor there will be no detection alert by this node. This is due to the method of calculating the threshold.

3. If a node exists with with all routes traversing only two neighbors the detection process will not be started. This is due to the method of calculating the threshold.

3.5.3 Additional Cases Investigated

In the case analyzed above we showed that one attacker acting as a sinkhole can be identified in the stationary grid network of thirty nodes. Since the sinkhole was alerted upon by all but the sinkhole itself and one additional node, the chance of sinkhole discovery was 93% (based on the number of nodes that are alerted to a sinkhole in the network by the detection process divided by the number of nodes in the network - in this case, $28/30 = 93\%$). This is because the effect of the sinkhole extends through the network for routes at least five nodes long. The following cases investigate whether more than one sinkhole can be identified in the network of thirty nodes.

3.5.3.1 No Sinkholes in the Network

In the case of no sinkholes in the network, we would like to have no alerts, and to find no suspects. However, since no network is homogenous (including the OPNET MANET network set up as described), it is likely that there will be at least one node in a neighborhood of nearly homogenous transmitting neighbors that will be falsely identified as a sinkhole. Using the algorithm described based solely on one standard deviation as the threshold, in a network with no intentional sinkholes placed, there were twenty-six nodes that were alerted to the possibility of a sinkhole. Four false positives were identified. This result portends the possible result that after identified true positives (sinkholes) in a network, the process may then find one or more false positives.

3.5.3.2 Two Adjacent Sinkholes in the Network

In this case, we have included two adjacent sinkholes operating in the same area of the network (nodes V, W). Experimental results show that all but one node was alerted to the possibility of a sinkhole in the network, and nine nodes have two neighbor reliability values that indicate the presence of one or more sinkholes. Since the sinkholes are adjacent, the identification process first finds the sinkhole with the lowest value. After the identified sinkhole is removed from the network, as well as all routes in route lists incorporating the sinkhole, the remaining nodes automatically re-calculate the neighbor reliability values. The second sinkhole is identified by the identification process since it now has the lowest neighbor reliability value in the network.

In this instance, node DD again was not alerted to the possibility of a sinkhole. This time node DD had one neighbor reliability value that was high because it had no routes with sinkholes included. The other two neighbor reliability values were nearly equal, lower, but fairly close to the higher value. Therefore, the method of identification and alerting based on incorporating the standard deviation did not indicate an outlier based on our thresholding scheme.

Table 3.4 Neighbor Lists for Nodes K and L

Starting Node	Neighbors	Number Routes with Sinkholes vs. Number Routes	Neighbor Reliability Value
Node K Neighbors	F	1/3	3.999E-05
	G	2/6	3.882E-05
	L	3/9	3.958E-05
	P	3/8	3.885E-05
	Q	3/9	4.073E-05
Node L Neighbors	F	0/2	5.924E-05
	G	2/4	2.961E-05
	H*	3/4	1.592E-05
	K*	2/3	2.258E-05
	M	1/11	5.435E-05
	P	0/1	5.886E-05
	Q	3/8	3.888E-05
	R	1/8	5.245E-05

3.5.3.3 Two Randomly Distributed Sinkholes in the Network

There are two sinkholes in the network located near opposite corners of the grid (nodes I, V). Experimental results show that all but two of the other grid nodes were alerted to the presence of at least one sinkhole in the network by their neighbor reliability values. After applying the sinkhole identification process as described in this paper, each sinkhole was correctly identified by the alerted nodes.

Two of the nodes each have two neighbor reliability values that indicate the existence of one or more sinkholes. The first node, S, has two neighbors whose reliability values cross the threshold; both of the neighbors' reliability scores are affected by their routes through the same sinkhole. The single sinkhole was correctly identified.

The second node (L) also has two neighbors (H, K) with reliability values that indicate the presence of a sinkhole, as denoted by the asterisks in the Table 3.4. However, each neighbor's values are affected by routes through a different sinkhole. Therefore, if we follow the identification process in which we first place and query the neighbor with the lowest reliability value, one sinkhole is identified by node L. After the first identified sinkhole and all associated routes are removed from the network route lists, and the neighbor reliability values are re-calculated, the second sinkhole is identified by node L. However, after the associated routes are removed and the process followed a third time there were two false positives identified in the network.

The two nodes that gave null results (did not indicate the presence of a sinkhole) were K and DD. We would expect nodes that have no routes through the sinkholes to be unable to alert upon the possibility of a sinkhole. However, both of these nodes have routes traversing each sinkhole. The neighbor reliability values did not alert these nodes because none of the values crossed the calculated threshold for the node. Investigation shows that in both cases, the lists of routes through every immediate neighbor included nearly half of the routes traversing a sinkhole, and the other half of the routes not traversing a sinkhole. Therefore, the mean of the route set for each neighbor of a node falls between the values reported for routes with no sinkhole, and the values for routes with a sinkhole (Section 5.2.1 number 4). (It should also be noted that node K is at the edge of the network, and node DD is in a corner of the network. Although this placement does not guarantee failure in identifying a sinkhole, it does lower the possibility due to the smaller number of neighbor nodes for comparison. However, the success or failure also depends upon the routes contained in the routing table of the node.)

The result is no values fall outside of the threshold, and the node does not receive an alert. In essence, by the perspective of these two nodes, routes including sinkholes are as common in the network as routes with no sinkholes, and therefore will not be identified as outliers. This argument foretells the results we will see in future cases with increased numbers of sinkholes.

The Table 3.4 shows the neighbors of nodes K and L with their neighbor reliability values and the ratio of the number of routes that traverse the neighbor and a sinkhole to the routes that do not go through a sinkhole. In this case, node K has no neighbors for which the number of routes through a sinkhole is greater than the number of routes with no sinkhole nodes. L has two such neighbors - H and K (note that here we are not looking at node K, but as node K *as a neighbor of* node L). Nodes H and K both met the conditions we set for alerting node L of a possible sinkhole in the network. Node K did not receive any such alert.

3.5.3.4 Multiple Randomly Placed Sinkholes in the Network

Multiple randomly placed sinkhole nodes were placed in the network. Table 3.5 shows the number of sinkholes, alerts, sinkholes identified, sinkholes missed, and false sinkholes identified in the thirty node network. It is important to note that this table represents the number of node members that would alert upon a sinkhole and start the discovery process. However, with a protocol in place to determine which alerted node would take action, the other nodes would not initiate the identification process. As a consequence, the number of falsely identified sinkholes appears quite high. This is because several nodes would alert upon the same false node.

Table 3.5 shows that as the number of sinkholes increased, the number of nodes receiving alerts decreased. This was expected, because as sinkholes become more numerous, using the algorithm based on the standard deviation, localized neighborhoods of the network will see the results of the sinkholes as a norm, rather than an anomaly.

In this network we were able to identify all of the sinkholes in networks with six or less sinkholes. In the cases of more than six sinkholes, at least one sinkhole was left unidentified. After the sinkholes were identified, the sinkholes and associated routes were removed from the routing tables and the neighbor reliabilities re-calculated. The identification process was repeated until there were no alerts in the network, or until any alerts were deemed non-productive because of loops in the process (i.e. node A points to node B as the neighbor with the lowest neighbor reliability value, who points to node C, to node D, to node E, to node A). In all of the networks with 8 or fewer actual sinkholes randomly placed in the network at least one false positive was identified.

3.5.3.5 New Criteria for Threshold

The original threshold criteria allowed too many false sinkholes to be identified. Such identification could result in friendly nodes being excluded from the network. After reviewing the data for every case, an additional threshold was added to the algorithm. This threshold excluded nodes with neighbor reliabilities that were not at least one order of magnitude less than the average of the local neighbor reliabilities from causing alerts. The results seen in Table

Table 3.5 One Standard Deviation as Threshold

Sinkholes In Network	Alerts	Sinkholes Identified	Sinkholes Missed	False Sinkholes
0	26	0	0	4
2	28	2	0	2
4	25	4	0	$\geq$1
6	20	6	0	$\geq$1
8	10	7	1	$\geq$1
10	6	6	4	0
15	2	6	9	0
18	0	0	18	0

3.6 show that there were no false positives reported when using the new threshold. For the networks with four sinkholes or less, all of the sinkholes were properly identified. For networks with six, eight, or ten sinkholes, up to two additional sinkholes were unidentified when compared to the results in the Table 3.5. Since it is expected that it is highly unlikely there will be more than one or two misbehaving nodes in a network of thirty nodes, the tradeoff when using the enhanced threshold criteria is for less disruption in the network due to the removal of innocent nodes.

Table 3.6 Experimental Results - Additional Threshold Criteria

Sinkholes In Network	Alerts	Sinkholes Identified	Sinkholes Missed	False Sinkholes
0	0	0	0	0
2	13	2	0	0
4	14	4	0	0
6	12	5	1	0
8	8	5	3	0
10	3	4	6	0
15	1	0	15	0
18	0	0	18	0

3.6 Analysis of Network Overhead

The method presented does not depend upon stored patterns, signatures, or rules. It does, however, require limited storage (s) of additional collected data and a small amount of energy (e) for calculations. Messages (m) are only required for identification when a sinkhole is detected. Therefore, the overhead o can be described by the equation:

$$o = s + e + m. \tag{3.10}$$

3.6.1 Storage

The AODV routing tables already house information about the "next hop" to known destinations. The "next hop" is an immediate neighbor. Therefore, the data we require to be stored in the routing table can be associated with the "next hop", or neighbor. We assign s_s as the storage space used to hold the number of packets sent and s_r as the storage space for the number of packets received. After the calculation for the neighbor reliability is completed, it is stored in s$_{nr}$. The additional storage space s required by a network of i nodes, each with b neighbors, can therefore be represented by the equation

$$s = ib(s_s + s_r + s_{nr}). \tag{3.11}$$

3.6.2 Energy

The detection and identification processes rely upon the calculation of the packet delivery ratio and neighbor reliability values assigned to the b neighbors of the i nodes in the network. The energy required for the packet delivery ratio calculation is represented by e_{pd}. Similarly, the energy required by the neighbor reliability calculation is represented by e_{nr}. Additional energy is required by the creation (e_{mc}) and distribution (e_{md}) of the identification messages m. The total additional network energy consumption e required by the detection and identification processes can therefore be described by the equation

$$e = m(e_{mc} + e_{md}) + ib(e_{pd} + e_{nr}). \tag{3.12}$$

3.6.3 Messages

Messages related to the identification process are generated and transmitted in the network only when a sinkhole is detected. As mentioned before, there are no additional network messages required for the detection process. The message overhead is dependent upon how far, or the number of hops h, the detecting node is from the sinkhole. Therefore, m_{Low} refers to the number of messages required for the assignments of L_{Low} along the path to the sinkhole. Also partially dependent upon the number of hops to the sinkhole are the number of neighbor list messages (m_{nl}) which are sent by nodes that have both the assigning node and L_{Low} in common. Whether the sinkhole participates in the process by assigning a L_{Low} greatly affects the number of messages generated. We therefore apply binomials j and k to reflect participation where $j = 1$ for no participation, and $k = 1$ if there is participation. The additional messages generated near the sinkhole when the sinkhole fails to participate is m_f, with m_p reflecting the number of messages generated during the "splash back" beyond the sinkhole due to its participation. Finally, we include the messages generated by the average number of "voters", m_v . The number of messages required by the identification method can be approximated by the following equation:

$$m = m_{Low} + m_{nl} + km_p + jm_f + m_v. \tag{3.13}$$

This equation can be simplified by including the parameters for the number of hops and the average number of neighbors common to L_{n-1} and L_n in each hop to:

$$m = h + 2h + 5k + 3j + 3 \tag{3.14}$$

or

$$m = 3h + 8k + 6j. \tag{3.15}$$

Figure 3.5 provides a graph depicting the actual messages sent by nodes on each ring compared to the estimated number of messages for the ring. A protocol for sharing the burden of starting the sinkhole identification process would need to be developed in a real network, and is not in the scope of this paper. With the proper protocol, only one node would start the identification process, limiting the number of messages as overhead.

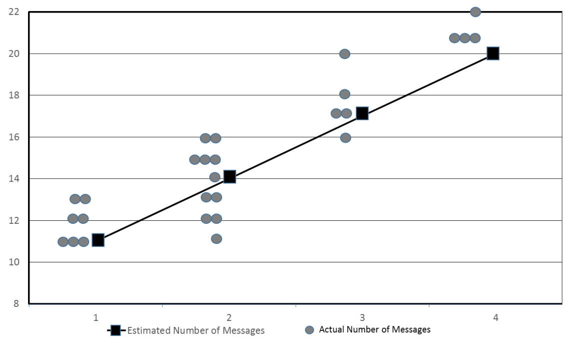

Figure 3.5 Number of Messages Per Ring

3.7 Conclusion

In this paper we presented a hypothesis that, by adapting a methodology borrowed from the science of meteorology, we can utilize the data available at both the node and cooperative network levels to create a synoptic picture of the network health, providing indications of any intrusions or other network issues. Our major contribution is to provide a revolutionary way to analyze node and network data for patterns, dependence, and effects that indicate network issues at a distance. This method did not rely upon the conventional methods of stored patterns for comparison, but only proper analysis of a subset of the real time parameters of the network.

Simulations using the network described in Table 3.1 showed that the original scheme found false sinkholes in networks with eight or fewer sinkhole nodes. However, the addition of a second threshold resulted in the elimination of the false positives, with the trade off of increased false negatives in networks of six or more sinkholes. This tradeoff is justified because we can realistically expect a network to have fewer than six (sinkhole) attackers. Therefore, using the two thresholds provides proper intrusion detection for this type of attack.

Our results showed the number of nodes in a thirty node network that would start the identification process and identify the sinkhole(s) based upon the threshold criteria. For the sinkhole identification process to be implemented in a real network, a protocol for sharing the responsibility of identifying the sinkhole would need to be developed. The objective of the protocol would be to allow identification of the sinkhole while limiting the number of nodes starting the identification process, and the number of messages flooding the network. Possible strategies are a round robin system based upon time, or assignment of responsibility to cluster heads.

The synoptic analysis technique presented in this paper is founded upon comparing the counts of events in or effects on the wireless network. Other attacks that are based upon causing or affecting countable events that trigger changes in network characteristics are candidates for synoptic analysis. Attacks at the physical, network, and data link layers such as jamming, HELLO flood, and wormhole assaults are likely contenders. Challenges to the use of the technique described include the development of protocols to identify the initiating node(s) and the development of a proper network reaction. An additional challenge is a method to provide the network nodes with more distributed network data without increasing the controlling network traffic.

CHAPTER 4. A METHOD FOR SYNOPTIC LEVEL NETWORK INTRUSION DETECTION IN A WIRELESS AD HOC NETWORK

A paper submitted to *Ad Hoc Networks*

Deanna T. Hlavacek[1][2][3] and J. Morris Chang[4]

4.1 Abstract

Current system administrators are missing intrusion alerts hidden by large numbers of false positives. Rather than accumulation more data to identify true alerts, we propose an intrusion detection tool that effectively uses select data to provide a picture of “network health”. Our hypothesis is that by utilizing the data available at both the node and cooperative network levels we can create a synoptic picture of the network providing indications of many intrusions or other network issues. Our major contribution is to provide a revolutionary way to analyze node and network data for patterns, dependence, and effects that indicate network issues. We collect node and network data, combine and manipulate it, and tease out information about the state of the network. We present a method based on utilizing the number of packets sent, number of packets received, node reliability, route reliability, and entropy to develop a synoptic picture of the network health in the presence of a sinkhole and a HELLO Flood attacker. This method conserves network throughput and node energy by requiring no additional control messages to be sent between the nodes unless an attacker is suspected. We intend to show that, although the concept of an intrusion detection system is not revolutionary, the method in which we analyze the data for clues about network intrusion and performance is highly innovative.

[1]Graduate Student, Department of Electrical and Computer Engineering, Iowa State University.
[2]Primary Researcher and Author.
[3]Author for correspondence.
[4]Associate Professor Department of Electrical and Computer Engineering, Iowa State University.

4.2 Introduction

Wireless ad hoc networks are self-organizing and self-configuring infrastructure-less networks of nodes which are connected by wireless links. Ad hoc networks are becoming more popular with 802.11/WiFi capable products for communication, entertainment, work, and study. Wireless sensor networks (WSNs) are being deployed in the home and office for security and energy conservation. WSNs are also deployed in many places in the environment, acting as watchdogs in forests to watch for fires, on mountaintops to alert on avalanches, and on bridges to report ice development. Additionally, the new cognitive radio is being developed with ad hoc capability. The cognitive radio provides enhanced environmental awareness and cooperative capabilities, and is capable of identifying and utilizing unused frequencies dynamically. This allows more concurrent wireless communications in a given spectrum band at one location. The cognitive radio is expected to be offered to the public, but its most anticipated deployments are to emergency responders and military forces to allow communication in infrastructure-less areas.

With the growth in popularity of these technologies, there is a growing demand for intrusion detection systems that can operate with network node cooperation. An intrusion detection system (IDS) is a software application that monitors the network for, and reports on, malicious activities or policy violations. Current research on intrusion detection systems for wireless ad hoc systems focuses mainly on anomaly detection. Generally, the anomalies garnering the most attention are anomalies in routing protocol, traffic patterns, and packet meta-data. However, anomaly and large data based intrusion detection systems are susceptible to high false positive rates. With limited time and resources, many true positives are lost in the overload of the combined true and false alerts, and other data.

An example of this is the Target point-of-sale breach in 2013. Although an alert was sounded by the intrusion detection system FireEye, system administrators missed the warning. Over 40 million credit card numbers were sent to Russia before administrators investigated the alarm and took action to close the breach [107]. Similar breaches occurred at Home Depot and a variety of restaurants and other retail centers. Since the false positive rate for anomaly detection tends to be high, research has now moved more towards hybrid solutions, combining anomaly detection with misuse detection or specification deviation. All of these methods require some prior training of the network nodes or pre-positioning of data for comparison.

Perhaps the solution is not more data. Rather, the solution may lie in the ability of a system to effectively use select data. Igor Baikalov, chief scientist at Securonix, was quoted in an article by the New York Times [107]: “We don’t need ’big data’. We need big information.” By carefully removing the noise created by large amounts of data, we allow our security professionals to focus on information with value, quickly identify attacks, and make timely decisions.

Our motivation for research related to intrusion detection arises from the current lack of comprehensive research into methods of analysis of selective information in an effort to construct a big picture of network security and integrity, termed as “network health”. Research into the parameters of the nodes and networks, the interplay of parameters and their effect upon each other, and how the concurrence of certain parameter levels portend negative or positive network health can bring valuable insight into the diagnosis of network ills.

Our hypothesis is that, by adapting a methodology borrowed from the science of meteorology, we can utilize the data available at both the node and cooperative network levels and create a synoptic picture of network health, providing indications of any intrusions or other network issues. Parameters such as packet delivery ratio, packet sequence number, route-add ratio, and many others have previously been used to alert on and/or identify intruders. However, this data also provides valuable information about the state of the network as a whole. By analyzing the packet, route, and node data at a network level we expect to develop a synoptic picture of the network and identify indicators comprised of different types and levels of data. The visual representation of the synoptic network picture is expected to be much like synoptic weather charts depicting the temperature, pressure, and relative humidity centers that, once properly analyzed, are indicative of changing weather. And, just as a meteorologist collects, combines, analyzes, and interprets temperature, pressure, humidity, wind direction, and wind speed to determine the weather conditions in an area, we can collect, combine, analyze, and interpret the number of sent packets, received packets, control messages, broken links, re-transmitted packets, and node energy consumption (for example) to paint a picture of the wireless network and determine the network health. In this sense, just as thermometers, barometers, and anemometers can monitor the environment for approaching storms, we have network tools that monitor parameters that can be used to identify malfunctioning areas of the wireless network. Since the synoptic analysis technique presented is founded upon comparing the counts of events in or effects on the wireless network it is anticipated that other attacks that are based upon causing or affecting countable events that trigger changes in network characteristics are candidates for synoptic analysis. Examples are attacks at the physical, network, and data link layers such as a sinkhole, jamming, HELLO flood, and wormhole assaults.

Our major contribution is to provide a revolutionary way to analyze node and network data for patterns, dependence, and effects that indicate network issues. Our first contribution in this vein was presented in [41]. It described a method based on utilizing packet delivery ratio (PDR), node reliability, route reliability, and entropy to develop a synoptic picture of the network health in the presence of a sinkhole. Future study will determine if the method, possibly enhanced, will work in a mobile network. This work revisits sinkhole detection and identification in a grid network as a demonstration. The work is expanded to include a sinkhole in a thirty node scrambled network. We also include the detection and identification of a HELLO Flood attacker using the same methodology. We intend to show that, although the concept of intrusion detection is not revolutionary, the method in which we analyze the data for clues about network intrusion and performance is innovative, and can be a valuable addition to the intrusion detection "toolbox".

In this paper we will take a look at current research related to intrusion detection systems designed for wireless ad hoc networks, sinkhole detection, and HELLO Flood attacker detection. Next we provide a description of our methodology in network analysis for detection and identification of a single sinkhole and multiple sinkholes in a grid network, and our results based on the simulation data as presented in [41]. Additionally we provide analysis of the technique based on a single sinkhole in a scrambled network and on a HELLO Flood attacker in a scrambled network. Last we will conclude with a synopsis of the process and the impact of our experimental results. The sections of the paper are organized as follows: Section 4.3.2 describes current research into intrusion detection systems, sinkhole detection, and HELLO Flood detection in wireless ad hoc networks; Section 4.4 presents the methodology of identifying an attacker based on the PDR, node reliability, and system entropy; Section 4.5 presents our results with simulation of a sinkhole and HELLO Flood attacker in wireless ad hoc networks, and data analysis; Section 4.6 provides comparison of the simulation results to other sinkhole and HELLO Flood attacker identification methods; and Section 4.7 is the conclusion.

4.3 Related Work in Wireless Ad Hoc Networks

Wireless ad hoc networks have no router or access point providing infrastructure to the network. Each node provides routing services, via routing protocols, by forwarding packets to their neighbors. All nodes in an ad hoc network have equal status in the network, and can associate with any network device within range. There are three main routing protocols for ad hoc networks: Optimized Link State Routing (OLSR), Dynamic Source Routing (DSR), and Ad Hoc On Demand Routing (AODV). OLSR is a proactive, or table-driven, protocol. DSR and AODV are both on-demand, reactive protocols in which the nodes maintain routing tables. AODV's routing tables are refreshed according to a timer. We have chosen to use the AODV routing scheme for our first demonstration.

4.3.1 Intrusion Detection Systems

Conventional intrusion detection systems (IDS) are based on misuse detection, anomaly detection, or deviation from specifications. Misbehavior/misuse detection refers to identifying an attack by comparing collected information against a predefined list of "signatures" of known attacks. Anomaly detection is a close opposite to misuse detection. With anomaly detection, rather than storing a list of the signatures of known attacks, the system stores patterns of "normal" behavior for comparison to the current behavior. The third technique, specification comparison, also compares the current behavior to a stored behavior profile. However, the comparison is against manually defined specifications, rather than machine learning and training techniques.

Recently, most of the intrusion detection system research for wireless ad hoc systems has focused upon the detection of anomalous behavior patterns. In the paper [110] a new detection scheme called AODVSTAT is presented. The method is similar to other watchdog schemes in that the nodes watch the packet events and the meta-data in the packets for anomalies in the protocol. Deviations from the protocol are considered state changes, and trigger an alarming event. In an effort to lower false positive rates using packet features as the basis for anomaly detection, the authors of [58] conduct careful feature selection from the available set of packet features for their method. Similarly, [53] uses the entropy of packet features to detect deviations, and provides a ranking of alerts in an attempt to lower the false positive rate. The authors of [108] base their system solely upon the packet sequence number mismatch with the expected packet sequence number. A confidence level for each node is calculated based on the number of interactions with its neighbors. Rather than packet features, the authors of [135] base their detection on anomalies in traffic patterns, comparing the current traffic pattern to a learned traffic pattern. All of these techniques either requires knowledge stored in memory of the "normal" pattern of behavior, or some type of training before deployment. Attacks that do not register against the learned normal profile can not be detected by these systems. These methods are also prone to false positives, since hiccups in the network can cause the systems to react with identification of pattern anomaly.

The authors of the papers [3, 34] both present hybrid detection schemes. The paper [3] combines anomaly and misuse detection schemes in order to lower the false positive rate generally seen with anomaly detection, and raise the low detection rate ascribed to misuse detection. The authors of [34] chose to combine the anomaly and specification based schemes, and uses a reputation based system in an attempt to lower the false positive rate. However, this method adds much packet overhead as the nodes in the neighborhoods vote.

In comparison to these methods of intrusion detection, our intrusion detection method is based not on stored patterns, signatures, or rules, but the effect upon the node, route, and network function. Our system requires no training or pre-placed data concerning the expected network protocol or traffic pattern. Therefore, new or craftily tweaked older intrusion methods, such as a stealthy sinkhole that follows network protocols while selectively dropping packets, will still be identified, as long as the effect of the disruption surpasses the established threshold. We have no watchdogs observing the network, and so we are not plagued by the high false positive rates endemic to these methods. Similarly, we are not doing signature comparison, so we are not plagued by the inability (and associated low accuracy rate) to recognize new attack signatures. Our system instead is monitoring the state of the network, and reacts when the state of the whole, or portions of, the network move out of alignment. As in holistic medicine, we do not only pay attention to the symptom; we analyze the symptoms to explore the network function and find the root cause of the malfunction.

4.3.2 Sinkhole Detection

We consider sinkholes as Byzantines in the network. Byzantine behavior is displayed by any action of a member node that negatively affects the routing service in the network. Many such attacks, such as eavesdropping or packet modification, can be prevented by traditional authentication, integrity, and encryption mechanisms. The malicious actions of a Byzantine sinkhole may be more complex, such as modifying the hop count, sequence number, or list of nodes in a path, in order to make itself more attractive as an entry to an ideal route. According to the paper [6], attacks using these tactics can also be prevented with more sophisticated authentication and integrity techniques. We therefore consider the stealthy sinkhole that drops data packets, entirely or selectively, while participating in the routing protocol.

The authors of [6] present a method of identifying a sinkhole with link weights and probes. However, this method is part of a newly proposed routing protocol that includes double flooding during route discovery and the sending of probes to all network nodes for attack discovery. These steps create additional network overhead. Additionally, in this routing protocol, the sinkhole will only be discovered if it is acting maliciously during the probing phase. Finally, this work provides no insight as to how to identify a sinkhole in the accepted ad hoc routing protocols Ad Hoc On Demand Distance Vector Routing (AODV), Dynamic Source Routing (DSR), or Optimized Link State Routing (OLSR).

In the paper [57] the authors present a packet drop attack detection method in which the neighbors adjacent to a communications route monitor the actions of the en route nodes. If a particular node does not forward a specific number of packets in a certain time period, an alert proclaiming a malicious node is started. The authors make a distinction between greyholes, which drop only a portion of the packets, and blackholes, which drop all received packets.

Analysis by the authors indicates that only blackholes, and gray holes in the same vicinity, can be identified. If no blackhole exists, the greyholes will not be identified. Similar to this method is the watchdog method presented by the authors of [39]. Again, collaborative nodes observe the actions of the nodes en route and use a protocol to determine when an alarm should be sounded. Unfortunately the reliability and effectiveness of this method is difficult to determine since the authors have not yet determined the false positive/false negative rates for the protocol.

Several papers rely upon the sinkhole bucking the routing protocol by changing the sequence numbers. The sinkhole identification schemes described in the papers [22, 33, 46] are therefore not effective in identifying a stealthy sinkhole that does not change the packet sequence number. The authors of [22] additionally use the previous image ratio to identify the sinkhole. In the previous image ratio method the received routing packets are compared to other stored routing packet images. However, this method relies upon the sinkhole having forged the route records in their route request packets. Therefore, if a stealthy sinkhole has not forged the route records, the sinkhole will not be identified by this method.

A third indicator of a sinkhole identified and used by the authors of [22, 46] is the route add ratio. The route add ratio is the number of routes that traverse a particular node divided by the total number of routes added to the node's routing table. Unfortunately, [46] only mentions the idea of the route add ratio, but does not explain how the ratio is used. In [22] a network node is specified to keep a counter for each node in the network, and increment the counter when a route passing through the node is added to the cache. This presents the issue of one node storing data for all of the nodes, and additional messages created and sent to the assigned node when any network node adds a route to its cache. The data related to this study did not provide a method to determine the message overhead related to this technique.

In comparison to these studies, our method does not rely solely upon the sinkhole cheating on the routing protocol. Therefore, a stealthy node will still be identified by its effect on the network, rather than missed due to it not sending signals through the routing protocol. Also, even though there are additional messages included with our sinkhole identification scheme, no additional messages are required before the possibility of a sinkhole is discovered. The overhead of messages m related to sinkhole identification is related to the number of hops from the detecting node to the sinkhole and is very small. We analyze the number of messages required in Section 4.6.3.

4.3.3 HELLO Flood Attacker Detection

The HELLO flood attack was first described by the authors of [51] as an attack against wireless sensor networks. However, due to similarities in the routing protocols of ad hoc networks, the attack can also be applied to the cognitive radio network [40]. The attack relies upon an attacking node flooding the network, or at least a portion of the network, with HELLO packets broadcast at a higher transmission power. Since the network nodes receiving the HELLO packets assume that the sender is within normal radio range, they will attempt to use the attacking node as a route to other nodes. This can result in a network in a state of confusion.

One counter measure against the HELLO flood attack suggested by the authors of [51] is to confirm the bidirectionality of a link using an identification verification protocol. The protocol employs an encrypted echoback mechanism before using the link. It is acknowledged that this defense is less effective when an attacker has a highly sensitive receiver along with the powerful transmitter.

The authors of [94] suggest the the network use a received signal strength (RSS) value as a threshold. Received HELLO packets are compared to the threshold. If the RSS of a packet is greater than the threshold, the node is determined to be a "stranger". If a packet is received such that the RSS meets the threshold, the sending node is determined to be a "friend". Network nodes additionally test "friend" nodes by sending a test packet. If the acknowledgment to the packet is not received from the "friend" in the allotted time, it is determined the "friend" is actually a "stranger".

In the paper [13] the authors suggest a solution using a base station. If the station determines there may be a denial of service (DoS) attack taking place, the station challenges clients with cryptographic puzzles. The difficulty of the puzzles distributed to a node is based upon a node's trust value.

A base station is also used as a Trusted Third Party in a countermeasure suggested in [51]. With this method, the base station facilitates the establishment of session keys between parties of the network. The keys are then used by the nodes to verify each other's identities. The number of shared keys must be limited so the attacker cannot establish a connection with every node.

In comparison to these studies, our method does not rely upon the use of a base station in the network. Additionally, the method proposed by the authors will not have the overhead of exchanging or storing cryptographic keys. Our method also will not depend upon the prior training of the network concerning the received signal strength threshold, nor will there be additional message overhead due to the sending of test packets. As will be shown, our method will only incur message overhead during the identification process after the HELLO flood attack is detected.

4.4 Methodology

The methodology of the attacker identification scheme consists of two parts. First, the detection phase, in which one or more nodes are alerted that there is a possible attacker in the network. Individual nodes calculate their neighbors' reliability values, and are alerted when a reliability crosses a threshold. The second phase is the attacker identification phase and involves querying specific network nodes for data they have about *their* neighbors. We make the assumption that each node is aware of its immediate neighbors. However, the nodes may not immediately be aware of the position of its neighbors relative to itself or each other.

4.4.1 Detection Process

In the neighbor reliability method of determining the health of the network, each node counts the number of packets sent and received along each route. The nodes calculate the packet delivery ratio for the stored routes by relating the number of packets received along the route to the number of packets sent along the route. This PDR is also referred to as the "route reliability", and represents the cumulative reliability for each node along a route. From this data, the nodes each determine their "neighbor reliability" by calculating the entropy for all known routes through the neighbor node. Neighbor reliability refers to a single node's perception of the probability of a packet following any route through a single neighbor to successfully reach

Algorithm 3 Calculations of Neighbor Reliability

```
//Done for each neighbor node (Y) around the starting node A
//routeRel: route reliability
//neighRel: neighbor reliability
1 : for each neighNode(Y)
//Calculate route reliability for routes X through neighNode Y
2 :   for each route(X) (Z nodes along route)
3 :     for each node(Z) on route X
4 :       routeRel(X) = routeRel(X) * pdr(Z);
5 :       saveRouteRel(XZ) = routeRel(X);
6 :       get next node pdr(Z) on route(X);
7 :    get next route(X);
//Calculate neighbor reliability using all routeRel(X)
8 :   for each routeRel(X)
9 :     neighRel(Y) = neighRel(Y) +
                  (routeRel(X) * [$\log_2$(1/routeRel(X))]);
10:     saveNeighRel(X) = neighRel(Y);
11:     get next routeRel(X);
12: get next neighNode(Y);
```

the intended destination. The Shannon entropy equation, used to estimate the diversity of the system, is applied. The formula follows, where p(x) is the route reliability:

$$H(x) = -\sum p(x)log_2p(x) = \sum p(x)log_2(1/p(x)) \tag{4.1}$$

Algorithm 3 describes the process to obtain the neighbor reliability values. Note that in simulation we use the probability that a packet will reach its destination when routed through an individual node for the PDR of node Z. This was substituted for the true PDR of a route that would be known in a live network.

Using this method, assuming a stationary MANET grid of nodes with one node per grid space, each node in the grid will have up to eight different neighbor reliability values, each value from the perspective of one of its (up to eight) neighbors. Additionally, each node will have up to eight neighbors for which it has determined neighbor reliability values, comprising the neighbor reliability set. Using its neighbor reliability set, each node calculates the standard deviation of the set. A lower boundary is calculated by subtracting the standard deviation from the mean of the set; the boundary acts as a threshold. The use of the standard deviation was determined experimentally; one standard deviation provided a proper boundary to determine if the neighbor reliability was low enough to indicate the possibility of a sinkhole when compared to the neighbor set. The lower boundary in a live network will need to be determined based upon the particular network. This process provides the node the ability to observe the current state of the network surrounding it, and helps identify anomalies based on the current network state. If the neighbor reliability of any neighbor crosses the threshold, the node is alerted that there may be an attacker in the network. Note that until this point, all calculations and decisions are made upon data collected by the node without additional messages or queries to the network or neighbors. Therefore, unless an alert is signaled, there is no impact upon the network throughput for this method.

The example in Figure 4.1 shows several nodes with reliability values applied to each node *by its neighbors*. The values in the figure are charted according to compass coordinates from the perspective of an individual node. For example, Node W applies the values of 5.443 to node AA, 5.324 to node BB, 5.436 to node CC, 5.276 to node X, 5.443 to node V, 5.393 to node R, 5.441 to node Q, and 5.329 to node S. These values comprise node W's neighbor reliability set. Conversely, node W is applied the values of 0.1322 as perceived by node AA, 0.1315 by node BB, 0.1331 by node C, 0.1333 by node X, et cetera.

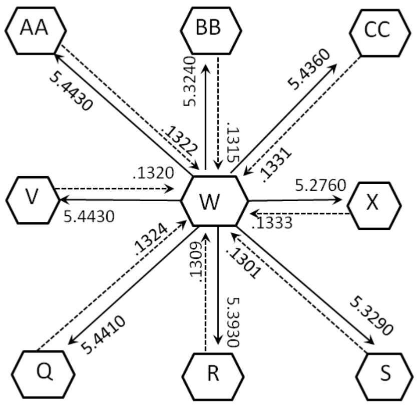

Figure 4.1 Neighbor Reliability Plot (values * 10e-6)

4.4.2 A Proof

We consider a simple grid network of nodes with a sinkhole placed in the center of the network (Figure 4.2). The sinkhole drops a large number of all received packets; let the percentage of successful packet delivery for the sinkhole be b. The network uses AODV routing, utilizing the shortest route. We make the following assumptions:

1. Each node is aware of all of its eight neighbors.
2. No route will pass through more than one of the source node's neighbors, and no more than two neighbors of any node along the route.
3. All routes are of three hops.
4. All nodes except the sinkhole drop a very small number of packets. Let the percentage of successful packet delivery for these nodes be a.
5. Percentage $a >> b$.

A source node (src1) located next to the sinkhole will have seven neighbors with routes that do not traverse the sinkhole, and therefore experience a packet delivery percentage of a. The last neighbor will have all packets routed through the sinkhole, experiencing a packet delivery percentage of b. The average percentage of packets experienced by this node will therefore be:

$$PDR_{src1} = (7a + b)/8. \tag{4.2}$$

A source node (src2) two hops from the sinkhole will have more routing choices that do not include the sinkhole. In the described network, five of the neighbors will have no routes through the sinkhole. Of the three neighbors left, two will each have four of their nineteen routes traversing the sinkhole (see Figure 4.2). Let this be represented by $PDR_{31/33.}$One neighbor will have three of its thirteen routes passing through the sinkhole (Figure 4.3). Let this be represented by PDR_{32}. The PDR experienced by src2 is therefore represented by the following:

$$PDR_{src2} = 5a + PDR_{31/33} + PDR_{32} \tag{4.3}$$

where

$$PDR_{31/33} = [2(15/19)a + 2(4/19)b]/8 \tag{4.4}$$

and

$$PDR_{32} = [(10/13)a + (3/13)b]/8 \tag{4.5}$$

or

$$PDR_{src2} = [7.53a + .65b]/8. \tag{4.6}$$

Likewise, a source node (src3) three hops from the sinkhole will have more routing choices that do not include the sinkhole. In this scenario seven of the neighbors do not have any routes through the sinkhole. One neighbor has three of its thirteen routes traversing the sinkhole. We get the following equation:

$$PDR_{src3} = [7a + (10/13)a + (3/13)b]/8 \tag{4.7}$$

or

$$PDR_{src3} = [7.77a + .23b]/8. \tag{4.8}$$

Since $a >> b$, we remove the terms with b from the equations. We can also remove the denominator of "8" since it is the same for all equations. By the relationship below, we prove that $PDR_{src1} < PDR_{src2} < PDR_{src3}$.

$$7a < 7.35a < 7.77a \tag{4.9}$$

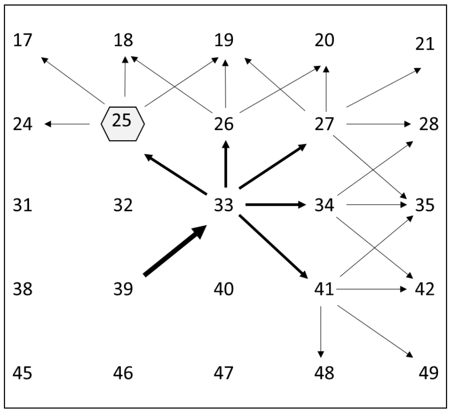

Figure 4.2 Route Example from SRC2 (Node 33)

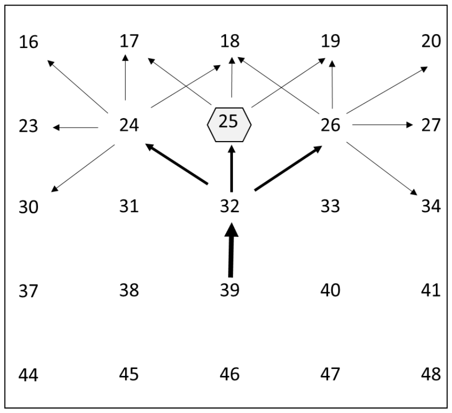

Figure 4.3 Route Example from SRC2 (Node 32)

4.4.3 Identification Process

The identification process works similar to water in a reservoir. When the water reaches a higher point in the reservoir, it stops moving forward and splashes back. In our identification process, the query messages will move towards the lowest reliability point and stop at the higher value nodes on the other side of the lowest point. Once the higher nodes are reached, the "splash" will be a broadcast message flooded through the network reporting the identity of the attacker. We use a sinkhole as the attacker for the following identification step description.

In the identification process, we assume that each node is aware of its immediate neighbors, although not necessarily their relative positions. We call this set of neighbors "ring one" as related to the alerted node. When a node determines that a threshold has been crossed, it broadcasts a query to its immediate neighbors (ring one) for their neighbor lists. The query includes the alerted node's assignment of L_1, which is the node with the lowest neighbor reliability from the perspective of the alerted node. Only the neighbors with neighbor lists that include L_1 broadcast a reply to the query. Node L_1 was able to receive the information sent by its own immediate neighbors in response to the query made by the alerted node. Node L_1 assigns L_2 to its neighbor with the lowest reliability value (L_2 will most likely, but not necessarily, be in ring two), and sends this assignment along with a broadcast query for neighbor lists. Only the neighbors with L_2 in their neighbor lists that have not already broadcast their neighbor lists during this identification process round reply to the query. (Algorithm 4)

Algorithm 4 Identification of Sinkhole

```
1 : initialize row, col, x to 0; AlertedNode is L0 at [0,0]
3 : Lx identifies Lx+1 as neighbor with the lowest neighbor value
4 : Lx broadcasts query for neighbor lists along with Lx+1 identification
5 : x=x+1
// if Lx participates in the discovery process it will automatically query neighbors
6 : while attacker not found
7 :   Lx identifies Lx+1 as neighbor with the lowest neighbor value
8 :   if Lx+1 == Lx-1 OR Lx+1 == Lx-2 AND if other shared neighbors that have not yet been polled exist
9 :     immediate neighbors (not Lx-2, Lx+1, or Lx-1) vote to determine which of the compared nodes has the lowest value
10:     node that is not determined lowest value alerts network of identity of attacker
11:   else if y > 2 AND Lx+1 == Lx-y
12:     Lx-1 removes Lx neighbor value from neighbor list
13:     x = x-1 //this is a loop with no positive attacker determination
14:   else if Lx queries neighbors for THEIR neighbor list
15:       Lx+1 receives neighbor lists from immediate neighbors and determines relative placement of immediate neighbors
16:  else if Lx-1 does not hear Lx query neighbors in time T
17:     Lx-1 queries immediate neighbors shared with Lx (Lshared_1 and Lshared_2) for their neighbor lists and neighbor values
18:     Lx-1 determines relative locations of neighbors shared between Lshared_1 and Lshared_2
19:     if neighbors shared include only Lx and Lx-1
20:       Lshared_1 and Lshared_2 send queries for neighbor lists and reliabilities
21:       neighbors with lists that include Lx respond to query
22:       Lshared_1 and Lshared_2 provide query information to Lx-1
23:     Lx-1 uses neighbor locations and neighbor values to determine if Lx has lowest local neighbor value
24:     if Lx has lowest neighbor value as reported by surrounding immediate neighbors
25:       Lx-1 alerts network Lx is a attacker
26:     else
27:       Lx-1 removes Lx neighbor value from neighbor list
28:       x = x-1
29:   x=x+1
30: end while
```

This process continues until we have reached the area around the sinkhole, ring n. The nodes queried in ring n will include the sinkhole and some of its immediate neighbors. At this point the sinkhole may or may not participate in the process. If it does participate, the sinkhole will continue the process by identifying L_{n-1}. L_{n-1} will then identify the sinkhole as its neighbor with the lowest reliability value. The sinkhole may not acknowledge, nor realize, it is the sinkhole. Therefore, the sinkhole (call the sinkhole S) assigns L_{Low} (call this node X) to one of its neighbors. Node X compares its neighbor reliability values and re-identifies the sinkhole (S) as its neighbor with the lowest neighbor reliability. To attain confirmation, X broadcasts a query to its neighbors for a "vote" as to which of the two nodes they identify as the node with the lowest value. Only nodes with both nodes S and X as neighbors reply. The node requesting the vote will flood the network with the identification message naming the sinkhole. If the sinkhole does not participate, the node (L_{n-1}) that sent the message to the sinkhole and its neighbors will note that there have been no messages from L_n in time T. At this point, L_{n-1} will resend a message to the sinkhole and neighbors common to the requesting node and the non-responding (possible) sinkhole, requesting the reliability values for all of their neighbors. If the neighbor lists received do not include at least three neighbors in common, additional queries will be sent by the neighbors of L_n to their neighbors. The additional information received will be provided to L_{n-1}. Analyzed together, this information will confirm that the node with the lowest reliability in ring n is indeed the sinkhole, and L_{n-1} will broadcast the message identifying the sinkhole to the network.

The action taken by the network upon identification of the attacking node is dependent upon the system protocol, and is beyond the scope of this paper. However, there are basically two types of actions: isolation of the node, or incentivation for proper network behavior. Isolation means the suspect node may be ignored by the network when it advertises access to routes to any destination, and so ostracized from the network. Incentivation means allowing the suspect node to send its own data only at the rate that it is providing to the network, and this allowed rate increases as the misbehaving node decreases the number of dropped packets and increases its packet delivery ratio. It must be recalled that although we have referred to the suspect node with low reliability an attacker, it may be a selfish node, or just a malfunctioning node. This method of identifying the unreliable node does not determine intent.

4.5 Simulation

We simulate the detection and identification in several networks. We start with a grid of thirty nodes with one sinkhole. We then simulate and provide analysis for up to eighteen sinkholes in the grid network. We also analyze a network of thirty nodes in a scrambled network with one sinkhole. Finally, we simulate a HELLO Flood attack in a scrambled network of twenty nodes. It is shown that the methodology described works for several attackers in one network, for scrambled and grid networks, and for both sinkhole and HELLO flood attackers.

Table 4.1 Network Parameters

Node Parameters	Value
Number of Nodes	30
Node Placement	6 by 5 grid
Simulation Duration	1500 seconds
Routing Protocol	AODV
Type of Stations	MANET
Node Speed	0 kts
Transmit Power	.0001 w
Packet Reception Power Threshold	-95 dBm
Buffer Size (Member Nodes)	256000
Buffer Size (sinkhole)	415
Route Length	5 nodes, 4 hops

4.5.1 Sinkhole Attack

To obtain simulated network data we chose OPNET Modeler 17.5, a commercially available tool set used by the communications industry for modeling, simulation, and analysis of communications networks and applications. The initial network topology is a six by five grid mobile ad-hoc network (MANET). The size of our network was chosen based on a study of the effects of insider attacks by the authors of [28]. According to the study, thirty nodes is the ideal size of network for a sinkhole to operate effectively.

For the initial study the nodes are all stationary and the parameters for the member nodes are uniform. Traffic supplied by the OPNET MANET model is used. Each node is a traffic generator. No specialized traffic is added, and no additional noise is added to the network. Ad hoc on demand (AODV) routing is chosen as the routing protocol. Route length for the identification method is limited to five nodes, with four hops. This limitation was imposed to provide a simple initial standard for comparing route reliability results. The routes used for the calculations are not all inclusive. Therefore, not all neighbors may play a part in determining the calculated neighbor reliability value. Parameters for the member nodes are shown in Table 4.12.

The buffer size for the member nodes was left at the OPNET MANET default of 256000 packets. The buffer size for the attacker was lowered to 415 packets to simulate a sinkhole.

This buffer size was chosen because it allowed packets to be dropped without stopping all traffic routing through the attacking node. The number of dropped packets at this setting impacted significantly the amount of data traffic sent by the attacking node. Normal nodes sent an average of 903.5 packets over the 1500 second simulation period, as opposed to the 17.3 packets sent by the sinkhole. Note that the 1500 second simulation period and the 15 second intervals were arbitrary. Since the sinkhole effect is recorded almost immediately and the effect is nearly constant over the entire period using the simulation parameters described, the interval parameters for recording the data can be optimized. Subsequent tests showed that the sinkhole could be detected in the first fifteen seconds of the commencement of network traffic. Data collected per node included:

1. At each node, the number of packets received per second in fifteen second intervals.

2. At each node, the number of packets sent per second in fifteen second intervals.

Table 4.2 Node G's Neighbor Reliability

Start Node	G					
Reliability Neighbor List	GA	GB	GC	GH	GL	GM
Reliability Value List	3.0541E-06	2.9556E-06	3.0578E-06	2.2118E-06	2.6235e-06	2.2152E-06
Reliability Cumulative (all nodes)	0.00025877					
Reliability Average (all nodes)	4.3128E-05					
Entropy Per Neighbor List	5.5955E-05	5.4287E-05	5.6014E-05	4.0733E-05	4.8238e-05	4.0790E-05

4.5.1.1 Discovery and Identification Processes Using Simulation Results

In this context, the packet sent ratio (PSR) is the ratio of the number of data packets sent to the number of data packets received at a particular node. To find the neighbor reliability in our simulation, we first calculate the route reliability by multiplying the PSRs for each node along a route, starting at the first hop (as opposed to the sending node). Next we calculate the entropy of all known routes through the neighbor node. Plotting the neighbor reliability values on a plot of the network shows that the lower values tend to be centered around the sinkhole.

For an example, we assume node G is the first node to identify a possible attacker. From the reliability calculations (Table 4.2), node G identifies neighbors H and M as having entropy values below the threshold value for node G. Node G currently has a view of the network that includes its immediate neighbors, although node G may not know the relative locations of the neighbor nodes at this time. Node G identifies node H as the neighbor with the lowest reliability value. Node G broadcasts this assignment and queries its neighbors for their neighbor lists.

Only G's neighbors that share H as a neighbor broadcast their neighbor list; therefore, nodes L, M, C, and B broadcast their lists. Upon receipt of the data, node H places the nodes on the grid relative to itself (4.4). Node H reports it's neighbor with the lowest neighbor reliability value to node M, and queries its own neighbors for their neighbor lists. Again, only the neighbors that share node M as a neighbor broadcast their neighbor lists. From this information, node M is able to place itself and its immediate neighbors on the grid. Node R is assigned as having the lowest neighbor reliability value by node M. Neighbors that share node R as a neighbor make their neighbor list reports. Node R identifies node W as the neighbor with the lowest reliability value, continues the process, and node W identifies node X. Node X then identifies node W as the neighbor with the lowest reliability. Since nodes W and X identified each other, shared neighbor nodes "vote" for the neighbor with the lowest value. Since W has participated in the process, and it has been identified as the neighbor with the lowest reliability in the local area, node W (or, if needed, node X) announces to the network W is the sinkhole.

We borrow a technique from weather forecasting to analyze the plot by drawing isopleths for the reader to indicate the levels of entropy in the network. This provides a human-readable

Table 4.3 Partial Table of Neighbor Reliability Values

Node H Neighbors	L	N	M
Reliability Values	4.73898E-05	5.32743E-05	3.51034E-05
Node M Neighbors	Q	R	S
Reliability Values	5.56492E-05	1.90244E-05	5.39725E-05
Node R Neighbors	V	W	X
Reliability Values	5.29084E-05	.130932E-05	5.31013E-05
Node W Neighbors	X	BB	CC
Reliability Values	5.27565E-05	5.32431E-05	5.43592E-05
Node X Neighbors	W	BB	DD
Reliability Values	.133259E-05	5.23300E-05	5.29058E-05

synoptic view of the network at the current time. As can be noted in Figure 4.4, the sinkhole has been identified as the lowest area of reliability in the network. Even though the immediate neighbors to the sinkhole node are not dropping packets, their proximity to the sinkhole affects their reliability value because of the number of routes they have stored for use in their routing table that traverse the sinkhole. Nodes farther away from the sinkhole have their reliability values less affected by the attacker because they tend to have fewer routes traversing the sinkhole.

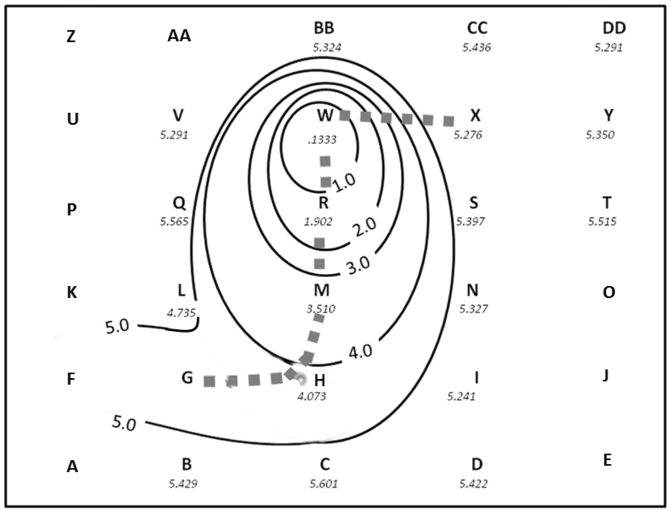

Figure 4.4 Synoptic analysis of neighbor reliability

4.5.1.2 Data Analysis - One Sinkhole

Analysis of the data set and the data related to each node shows that every node except node E resulted in an alert due to at least one of the neighbor nodes having entropy less than the one sigma lower boundary threshold. It must be noted that node E did not have any routes traversing node W (the sinkhole) in its routing table. Every other node had at least one route traversing node W. Also, starting the identification process from every node (except E) found W to be the attacker (or malfunctioning node).

There are cases in which the identification process may not be initiated and the sinkhole not found. However, as the network matures and nodes store routes with the sinkhole traversed, the identification process will be initiated. The special cases are listed below:

1. If a node exists with no routes stored that traverse the sinkhole, the sinkhole will not be detected by this particular node.

2. If a node exists with all routes traversing only one neighbor there will be no detection alert by this node. This is due to the method of calculating the threshold.

3. If a node exists with with all routes traversing only two neighbors the detection process will not be started. This is due to the method of calculating the threshold.

4.5.1.3 Additional Cases Investigated

In the case analyzed above we showed that one attacker acting as a sinkhole can be identified in the stationary grid network of thirty nodes. Since the sinkhole was alerted upon by all but the sinkhole itself and one additional node, the chance of sinkhole discovery was 93% (based on the number of nodes that are alerted to a sinkhole in the network by the detection process divided by the number of nodes in the network - in this case, $28/30 = 93\%$). This is because the effect of the sinkhole extends through the network for routes at least five nodes long. The following cases investigate whether more than one sinkhole can be identified in the network of thirty nodes.

No Sinkholes in the Network In the case of no sinkholes in the network, we would like to have no alerts, and to find no suspects. However, since no network is homogenous (including the OPNET MANET network set up as described), it is likely that there will be at least one node in a neighborhood of nearly homogenous transmitting neighbors that will be falsely identified as a sinkhole. Using the algorithm described based solely on one standard deviation as the threshold, in a network with no intentional sinkholes placed, there were twenty-six nodes that were alerted to the possibility of a sinkhole. Four false positives were identified. This result portends the possible result that after identified true positives (sinkholes) in a network, the process may then find one or more false positives.

Two Adjacent Sinkholes in the Network In this case, we have included two adjacent sinkholes operating in the same area of the network (V, W). Experimental results show that all but one node was alerted to the possibility of a sinkhole in the network, and nine nodes have two neighbor reliability values that indicate the presence of one or more sinkholes. Since the sinkholes are adjacent, the identification process first finds the sinkhole with the lowest value. After the identified sinkhole is removed from the network, as well as all routes in route lists incorporating the sinkhole, the remaining nodes automatically re-calculate the neighbor reliability values. The second sinkhole is identified by the identification process since it now has the lowest neighbor reliability value in the network.

In this instance, node DD again was not alerted to the possibility of a sinkhole. This time node DD had one neighbor reliability value that was high because it had no routes with sinkholes included. The other two neighbor reliability values were nearly equal, lower, but fairly close to the higher value. Therefore, the method of identification and alerting based on incorporating the standard deviation did not indicate an outlier based on our thresholding scheme.

Table 4.4 Node L's Neighbors Below 1 Sigma Threshold

Results	Neighbor	Neighbor
Entropy Outlier Name	LH	LK
Entropy Outlier Value	1.592E-05	2.258E-05

Table 4.5 Neighbor Lists for Nodes K and L

Node K Neighbors	Number Routes with Sinkholes vs. Number Routes	Neighbor Reliability Value	Node L Neighbors	Number Routes with Sinkholes vs. Number Routes	Neighbor Reliability Value
KF	1/3	3.999E-05	LF	0/2	5.924E-05
KG	2/6	3.882E-05	LG	2/4	2.961E-05
KL	3/9	3.958E-05	LH	3/4	1.592E-05
KP	3/8	3.885E-05	LK	2/3	2.258E-05
KQ	3/9	4.073E-05	LM	1/11	5.435E-05
			LP	0/1	5.886E-05
			LQ	3/8	3.888E-05
			LR	1/8	5.245E-05

Two Randomly Distributed Sinkholes in the Network There are two sinkholes in the network located near opposite corners of the grid (I, V). Experimental results show that all but two of the other grid nodes were alerted to the presence of at least one sinkhole in the network by their neighbor reliability values. After applying the sinkhole identification process as described in this paper, each sinkhole was correctly identified by the alerted nodes.

Two of the nodes each have two neighbor reliability values that indicate the existence of one or more sinkholes. The first node, S, has two neighbors whose reliability values cross the threshold; both of the neighbors' reliability scores are affected by their routes through the same sinkhole. The single sinkhole was correctly identified.

The second node (L) also has two neighbors with reliability values that indicate the presence of a sinkhole (Table 4.4). However, each neighbor's values are affected by routes through a different sinkhole. Therefore, if we follow the identification process in which we first place and query the neighbor with the lowest reliability value, one sinkhole is identified by node L. After the first identified sinkhole and all associated routes are removed from the network route lists, and the neighbor reliability values are re-calculated, the second sinkhole is identified by node L. However, after the associated routes are removed and the process followed a third time there were two false positives identified in the network.

The two nodes that gave null results (did not indicate the presence of a sinkhole) were K and DD. We would expect nodes that have no routes through the sinkholes to be unable to alert upon the possibility of a sinkhole. However, both of these nodes have routes traversing each sinkhole. The neighbor reliability values did not alert these nodes because none of the values crossed the calculated threshold for the node. Investigation shows that in both cases, the lists of routes through every immediate neighbor included nearly half of the routes traversing a sinkhole, and the other half of the routes not traversing a sinkhole. Therefore, the mean of the route set for each neighbor of a node falls between the values reported for routes with no sinkhole, and the values for routes with a sinkhole (Section 5.2.1 number 4). (It should also be noted that node K is at the edge of the network, and node DD is in a corner of the network. Although this placement does not guarantee failure in identifying a sinkhole, it does lower the possibility due to the smaller number of neighbor nodes for comparison. However, the success or failure also depends upon the routes contained in the routing table of the node.)

The result is no values fall outside of the threshold, and the node does not receive an alert. In essence, by the perspective of these two nodes, routes including sinkholes are as common in the network as routes with no sinkholes, and therefore will not be identified as outliers. This argument foretells the results we will see in future cases with increased numbers of sinkholes.

The Table 4.5 shows the neighbors of nodes K and L with their neighbor reliability values and the ratio of the number of routes that traverse the neighbor and a sinkhole to the routes that do not go through a sinkhole. In this case, node K has no neighbors for which the number of routes through a sinkhole is greater than the number of routes with no sinkhole nodes. L has two such neighbors - H and K (note that here we are not looking at node K, but as node K *as a neighbor of* node L). Nodes H and K both met the conditions we set for alerting node L of a possible sinkhole in the network. Node K did not receive any such alert.

Multiple Randomly Placed Sinkholes in the Network Multiple randomly placed sinkhole nodes were placed in the network. Table 4.6 shows the number of sinkholes, alerts, sinkholes identified, sinkholes missed, and false sinkholes identified in the thirty node network. It is important to note that this table represents the number of node members that would alert upon a sinkhole and start the discovery process. However, with a protocol in place to determine which alerted node would take action, the other nodes would not initiate the identification process. As a consequence, the number of falsely identified sinkholes appears quite high. This is because several nodes would alert upon the same false node.

Table 4.6 shows that as the number of sinkholes increased, the number of nodes receiving alerts decreased. This was expected, because as sinkholes become more numerous, using the algorithm based on the standard deviation, localized neighborhoods of the network will see the results of the sinkholes as a norm, rather than an anomaly.

In this network we were able to identify all of the sinkholes in networks with six or less sinkholes. In the cases of more than six sinkholes, at least one sinkhole was left unidentified. After the sinkholes were identified, the sinkholes and associated routes were removed from the routing tables and the neighbor reliabilities re-calculated. The identification process was repeated until there were no alerts in the network, or until any alerts were deemed non-productive because of loops in the process (i.e. node A points to node B as the neighbor with the lowest neighbor reliability value, who points to node C, to node D, to node E, to node A). In all of the networks with 8 or fewer actual sinkholes randomly placed in the network at least one false positive was identified.

Half the Network Consists of Randomly Placed Sinkholes This case is constructed to determine if sinkholes can still be identified when half of the network nodes are acting as sinkholes. The sinkholes were randomly placed in the network. Because half of the nodes are sinkholes, and half are not, the network as a whole can be considered homogenous. Therefore, we expect the difficulty on identifying a single sinkhole to be high.

Results of the experiment show that two nodes were alerted to the possibility of a sinkhole. One of node K's neighbors had all routes traversing at least 2 or more sinkholes. This resulted in a very low neighbor reliability value, and so node K received an alert. After several iterations in which a sinkhole was identified, the sinkhole and its associated routes were removed from the network and route lists, the neighbor calculations were recalculated, and another node was alerted, six out of the fifteen sinkholes were identified.

In the case of the second node that received an alert, node Z, all of the neighbor reliabilities were affected by routes through one or more sinkholes. However, two of node Z's immediate neighbors were sinkholes. The routes of one of these neighbors traversed enough other sinkholes to result in a neighbor reliability value that crossed the established threshold calculated by the standard deviation process. After following the process described in 4, node Z was identified as a sinkhole. After removing node Z and all routes from the tables traversing node Z, no nodes received alerts indicating the possibility of a sinkhole in the network.

It must be noted that the analysis has been conducted in a static network; i.e. no additional routes are being added to the route lists as the sinkholes and their associated routes are removed from the network. In an active network, nodes attempting to communicate would be replacing removed routes with new routes not including a sinkhole, thereby rebuilding their route lists. These results show that the method described in this paper is heavily route and route list dependent, and performs best in a mature network.

Table 4.6 Experimental Results - One Standard Deviation as Threshold

Sinkholes In Network	Alerts	Sinkholes Identified	Sinkholes Missed	False Sinkholes
0	26	0	0	4
2	28	2	0	2
4	25	4	0	$\geq$1
6	20	6	0	$\geq$1
8	10	7	1	$\geq$1
10	6	6	4	0
15	2	6	9	0
18	0	0	18	0

4.5.1.4 New Criteria for Threshold

The original threshold criteria allowed too many false sinkholes to be identified. Such identification could result in friendly nodes being excluded from the network. After reviewing the data for every case, an additional threshold was added to the algorithm. This threshold excluded nodes with neighbor reliabilities that were not at least one order of magnitude less than the average of the local neighbor reliabilities from causing alerts. The results seen in Table

4.7 show that there were no false positives reported when using the new threshold. For the networks with four sinkholes or less, all of the sinkholes were properly identified. For networks with six, eight, or ten sinkholes, up to two additional sinkholes were unidentified when compared to the results in the Table 4.6. Since it is expected that it is highly unlikely there will be more than one or two misbehaving nodes in a network of thirty nodes, the tradeoff when using the enhanced threshold criteria is for less disruption in the network due to the removal of innocent nodes.

4.5.1.5 Generalized results for grid networks with more than one sinkhole

Analysis of the data from the simulation has shown that under other than the special conditions listed above, the sinkhole will eventually be discovered. The intent of the sinkhole is not determined; the misbehaving node may be an attacker, a malfunctioning node, or a selfish node. Only the actions of the node in relation to the network identifies the node as a sinkhole. It should also be noted that the AODV periodically calls for the clearing of the routing tables. This provides a selfish or malfunctioning node the opportunity to re-enter the network.

Experiments with two, three, four, five, and six sinkholes present indicate that using the same method as described, each of the sinkholes can be individually discovered. However, several findings must be noted:

1. The identification process will only identify one sinkhole per originating node with process completion.

2. After removal of all routes traversing an identified sinkhole from the routing tables, if the discovery process warrants the commencement of the identification process, an additional sinkhole will likely be discovered. This will continue until no new alerts are received.

3. If two sinkholes are adjacent, the sinkhole with the lowest value will initially be discovered. Once the network has identified the sinkhole and all nodes have removed all routes from their tables that traverse the sinkhole, the discovery and identification phases can begin again. At this time the adjacent sinkhole will be discovered.

4. Nodes in different areas of the network alerted to a suspect sinkhole will identify (generally) the sinkhole that is closest to them. As mentioned in 2, with the removal of the routes

traversing the identified sinkhole, additional sinkholes will likely be found if the discovery process warrants the commencement of the identification process.

Table 4.7 Experimental Results - Additional Threshold Criteria

Sinkholes In Network	Alerts	Sinkholes Identified	Sinkholes Missed	False Sinkholes
0	0	0	0	0
2	13	2	0	0
4	14	4	0	0
6	12	5	1	0
8	8	5	3	0
10	3	4	6	0
15	1	0	15	0
18	0	0	18	0

4.5.1.6 Scrambled Network

A scrambled (Figure 4.5) network of thirty nodes with one sinkhole was simulated. Node S was assigned as the sinkhole. Besides the node placement, the network parameters are as provided by Table 4.12.

Results of the simulation show that twenty-five of the nodes were alerted to the sinkhole using the additional threshold criteria described in Section 4.5.1.4. Four nodes other than the sinkhole did not alert to the sinkhole. The nodes not receiving alerts were C, I, N, and L. Nodes I, N, and L each have only two neighbors; therefore, special case three as described in Section 4.5.1.2 applies. Node C has has three neighbors. However, two of its neighbors have a similar number of routes through the sinkhole, and the third neighbor does not have any traversing routes that cross the sinkhole. Consequently, an outlier is not observed and node C is not alerted to the presence of the sinkhole.

4.5.1.7 Effects of Network Size on Sinkhole Detection

The network size of thirty nodes was initially chosen based upon the study presented in [28]. Subsequent tests using the sinkhole detection process described show that in networks smaller than 30 nodes the sinkhole can be detected using the initial threshold of one standard deviation

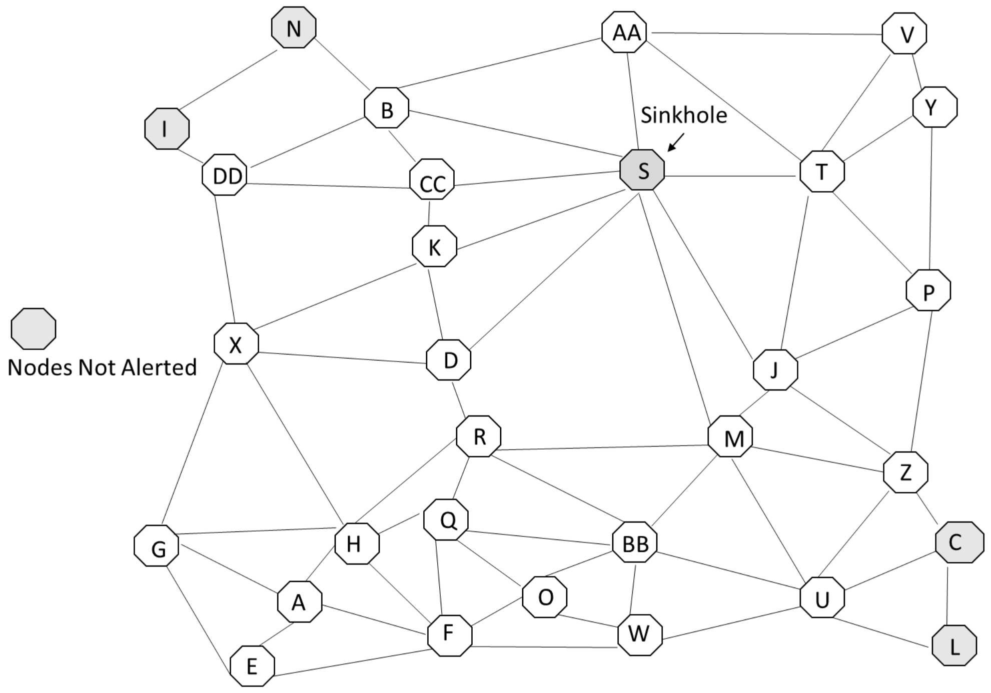

Figure 4.5 Scrambled Network

below the local neighbor reliability average. However, in the smaller networks the sinkhole is not detected with the additional requirement of a neighbor reliability value at least one order of magnitude smaller than its local neighbor's average reliability value.

Tests were conducted on a larger network with varying numbers of hops in the routes. A one-hundred node (ten by ten) grid network was created with the parameters displayed in Table 4.8. The sinkhole was placed in the top left corner of the large network, one node from the left edge and one node from the top edge of the network (a similar potion as node V in 4.4). Routes of four hops (five nodes), five hops (six nodes), and six hops (seven nodes) were simulated and the data analyzed for detection and identification of the sinkhole. "Percent accurately alerted" is the percentage of nodes that are within five hops of the sinkhole in the respective networks that actually detect the sinkhole. As expected, the total number of nodes that are within the route length limit of the sinkhole and can therefore detect the sinkhole does not include the sinkhole.

Comparison of results for routes of four hops, five nodes in the original thirty node network to results observed in the larger network are displayed in Table 4.9. In both networks all of

Table 4.8 Network Parameters

Node Parameters	Value
Number of Nodes	100
Node Placement	10 by 10 grid
Simulation Duration	200 seconds
Routing Protocol	AODV
Type of Stations	MANET
Node Speed	0 kts
Transmit Power	.0001 w
Packet Reception Power Threshold	-95 dBm
Buffer Size (Member Nodes)	256000
Buffer Size (sinkhole)	415

the nodes within four hops of the sinkhole were alerted to its presence. However, in the larger network there were a high number of false alerts. In order to reduce this number we added an additional criteria during analysis of the large network. The original additional criteria of one magnitude smaller (as described in 4.5.1.4) eliminated all of the true and false alerts in the large network. Instead we applied a new additional criteria. The new threshold is equivalent to one-half of one order of magnitude of the neighbor reliability subtracted from the whole neighbor reliability. Nodes not meeting this criteria are excluded from causing alerts. A comparison of the results of the large network analysis with and without the additional criteria is in Table 4.10.

As can be seen in Table 4.10, addition of the new criteria drops the number of false positives (incorrectly alerted nodes) from fifty-eight to one node. The trade-off is that the number of correctly alerted nodes drops from thirty-five to twenty-eight nodes. This means that the network would still be alerted to the presence of the sinkhole and the identification process would be started.

Table 4.11 presents the results of the five, six, and seven node length routes in the larger, one-hundred node network. Both the original results and the results when incorporating the new criteria are displayed. The results from the large network indicate that the original method presented for identifying one or more sinkholes in a thirty node network needs tweaking for differently sized networks. The new criteria we added to eliminate a number of the false positives

Table 4.9 Comparison of Sinkhole Discovery in Small and Large Networks (Four Hop Routes)

	Small Network	Large Network
Correctly Alerted (True Positive)	28 nodes	35 nodes
Correctly Did Not Alert (True Negative)	2 nodes	7 nodes
Incorrectly Alerted (False Positive)	0 nodes	58 nodes
Incorrectly Did Not Alert (False Negative)	0 nodes	0 nodes
Percent Correctly Alerted	100%	100%

without eliminating the true positives was dependent upon the size of the network. It is expected the requirement for criteria tweaking is due to the fidelity of the network data; in a larger network more variation between the PDR of nodes was observed, and this variation resulted in a larger number of false positives. However, after proper tweaking, the number of true positives remained high enough to ensure detection of the sinkhole with a small number of false positives.

4.5.2 HELLO Flood Attack

In order to simulate a network under attack by a HELLO Flooder we create a MANET consisting of twenty randomly placed nodes (see Figure 4.6). The area covered by the network is 20,000 square meters. The nodes are stationary and the parameters for the member nodes are uniform. Traffic supplied by the OPNET MANET model is used. Each node is a traffic generator. No specialized traffic is added, and no additional noise is added to the network. Ad

Table 4.10 Comparison of Sinkhole Discovery in Large Networks (Four Hop Routes) with Enhanced Threshold

	Without New Criteria	With New Criteria
Correctly Alerted (True Positive)	35 nodes	28 nodes
Correctly Did Not Alert (True Negative)	7 nodes	64 nodes
Incorrectly Alerted (False Positive)	58 nodes	1 node
Incorrectly Did Not Alert (False Negative)	0 nodes	7 nodes
Percent Correctly Alerted	100%	80%

hoc on demand (AODV) routing is chosen as the routing protocol. Route length for the identification method is limited to five nodes, with four hops. The routes used for the calculations are not all inclusive. Therefore, not all neighbors may play a part in determining the calculated neighbor reliability value. Parameters for the member nodes are shown in Table 4.12.

The transmission power of the member nodes was .0001 w. The transmission power of the attacking node was raised to .100 w; this allowed the attacker to transmit the HELLO packets to approximately half of the network. The packet reception power threshold was -95 dBm for all nodes. The number of dropped packets at this setting impacted significantly the amount of data traffic sent by the attacking node. The attacker was also close enough to the network to receive packets from some of the closest nodes. Normal nodes received an average of 63,958 packets over the 200 second simulation period, as opposed to the 3776 data packets received by the HELLO flooder. Note that the 200 second simulation period and the 15 second intervals were arbitrary. Data collected per node included:

Table 4.11 Results for Large Network - Five, Six, Seven Node Routes

	Five Node Routes - 35 Nodes in Range		Six Node Routes - 48 Nodes in Range		Seven Node Routes - 63 Nodes in Range	
	W/out New Criteria	With New Criteria	W/out New Criteria	With New Criteria	W/out New Criteria	With New Criteria
Correctly Alerted (True Positive)	35 nodes	28 nodes	47 nodes	34 nodes	61 nodes	46 nodes
Correctly Did Not Alert (True Negative)	7 nodes	64 nodes	10 nodes	48 nodes	8 nodes	30 nodes
Incorrectly Alerted (False Positive)	58 nodes	1 node	43nodes	5 node	30 nodes	8 node
Incorrectly Did Not Alert (False Negative)	0 nodes	7 nodes	0 nodes	13 nodes	0 nodes	15 nodes
Percent Correctly Alerted	100%	80%	98%	71%	97%	73%

1. At each node, the number of packets received per second in fifteen second intervals.

2. At each node, the number of packets sent per second in fifteen second interval.

4.5.2.1 Discovery and Identification Processes Using Simulation Results

To find the neighbor reliability in our simulation, we first calculate the route reliability by multiplying the PDRs for each node along a route, starting at the first hop (as opposed to the sending node). Next we calculate the entropy of all known routes through the neighbor node. Plotting the neighbor reliability values on a plot of the network shows that the lower values tend to be centered around the sinkhole.

Table 4.12 Network Parameters

Node Parameters	Value
Transmit Power (member nodes)	.0001 w
Transmit Power (attacker)	.100 w
Hello interval (member nodes)	(1, 1.1)
Hello interval (attack node)	(.1, .2)
Route Length	5 nodes, 4 hops

For an example, we assume node N is the first node to identify a possible attacker. From the reliability calculations (Table 4.13), node N identifies neighbor U as having an entropy value below the threshold value for node N. Node N currently has a view of the network that includes its immediate neighbors, although node N may not know the relative locations of the neighbor nodes at this time. Node N identifies node U as the neighbor with the lowest reliability value. Node N broadcasts this assignment to its neighbors, which includes node U.

Node U identifies node Q as the neighbor with the lowest reliability, and broadcasts this to the neighborhood. Since node M (the attacker) is broadcasting messages with greater power, node Q perceives node M as a neighbor. Node Q identifies node M as the neighbor with the lowest neighbor reliability value. Node Q broadcasts this to the neighborhood.

Node M is not truly a neighbor of node Q; therefore node M does not receive the message identifying it as the neighbor with the lowest reliability. Hence, node Q does not receive a response from node M in time T. Node Q rebroadcasts the identification of M and queries its neighbors for *their* neighbor with the lowest reliability value. Nodes R, P, and K also believe they share M as a neighbor, and M is their neighbor with the lowest reliability value. Node Q announces the identity of the attacker as node M.

Table 4.13 Node N's Neighbor Reliability (values X 10e6)

Starting Node: N	Cum. Rel.: 6115		Ave. Rel.: 1529	
Reliability Neigh. List	D	P	Q	U
Reliability Value List	6046	1994	3455	.000232
Entropy Per Neigh. List	196477	6474	1129325	.001825

4.5.2.2 Analysis

Analysis of the data set shows that seven nodes in the twenty node network did not detect the attack (see Figure 4.6). One node was the attacker itself. Five of these nodes were not within range of the HELLO messages and did not list routes that included the attacker (node M). Each of the twelve nodes in range of the attacker except node P alerted to at least one of their neighbor nodes having entropy less than the one sigma lower boundary threshold. It is noted that node P had a similar number of neighbors with routes through the attacker as neighbors without. This resulted in not enough variation in the reliability values to indicate an attacker. Node S was able to detect the attack; node S was sufficiently close to M that a few of the messages were received and forwarded by node M. Starting the identification process from every alerted node found M to be the attacker.

There are cases in which the identification process may not be initiated and the attacker not found. However, as the network matures and nodes store routes with the attacker included as a hop, the identification process will be initiated. The special cases are listed below:

1. If a node exists that does not receive the HELLO flood messages and has no routes stored that traverse the attacker, the node will not detect the attack.

2. If a node exists with all routes traversing only one neighbor there will be no detection alert by this node. This is due to the method of calculating the threshold.

3. If a node exists with with all routes traversing only two neighbors the detection process will not be started. This is due to the method of calculating the threshold.

Table 4.14 Partial Table of Neighbor Reliability Values (values X 10e6)

U's Neighbors	N	Q	T		
Rel. Values	6023	2418	6156		
Q's Neighbors	K	M	N	R	U
Rel. Values	2267	.000219	5462	.000234	5843

4. If a node exists that is within normal reception range of the attacker, and the attacker properly routes packets it receives, the node will not detect the attack.

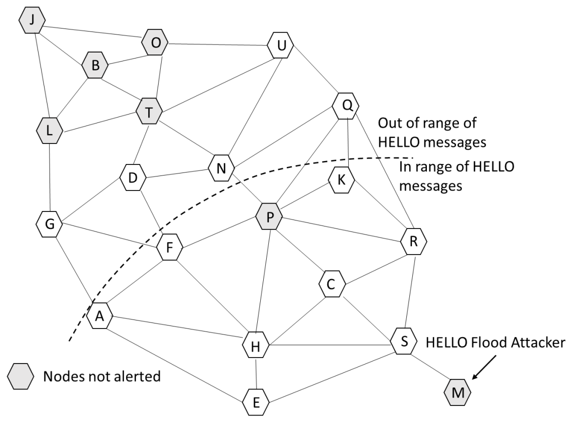

Figure 4.6 Simulated Network

4.6 Analysis of Network Overhead

The method presented does not depend upon stored patterns, signatures, or rules. It does, however, require limited storage (s) of additional collected data and a small amount of energy (e) for calculations. Messages (m) are only required for identification when an attacker is detected. Therefore, the overhead o can be described by the equation:

$$o = s + e + m. \tag{4.10}$$

4.6.1 Storage

In the detection method described we use the crossing of the neighbor reliability threshold to alert a network node to the presence of an attacker. The neighbor reliability value calculations depend upon the packet delivery ratios experienced when messages are sent through each neighbor. To determine the packet delivery ratio and the consequent neighbor reliability, each node needs to store the number of packets sent and the number of packets received when routing through each neighbor. This necessitates additional storage in the routing tables warehoused at each node.

The AODV routing tables already house information about the “next hop” to known destinations. The “next hop” is an immediate neighbor. Therefore, the data we require to be stored in the routing table can be associated with the “next hop”, or neighbor. We assign s_s as the storage space used to hold the number of packets sent and s_r as the storage space for the number of packets received. After the calculation for the neighbor reliability is completed, it is stored in s$_{nr}$. The additional storage space s required by a network of i nodes, each with b neighbors, can therefore be represented by the equation

$$s = ib(s_s + s_r + s_{nr}). \tag{4.11}$$

4.6.2 Energy

The detection and identification processes rely upon the calculation of the packet delivery ratio and neighbor reliability values assigned to the b neighbors of the i nodes in the network. The energy required for the packet delivery ratio calculation is represented by e_{pd}. Similarly, the energy required by the neighbor reliability calculation is represented by e_{nr}. Additional energy is required by the creation (e_{mc}) and distribution (e_{md}) of the identification messages m. The total additional network energy consumption e required by the detection and identification processes can therefore be described by the equation

$$e = m(e_{mc} + e_{md}) + ib(e_{pd} + e_{nr}). \tag{4.12}$$

4.6.3 Messages

Messages are only required in the network if an attacker is detected. The number of messages required by the method in the case of sinkhole detection is generally larger than the number of messages required in the case of a HELLO Flood attacker. This is because the HELLO Flood attacker broadcasts its HELLO message with greater energy, so many nodes in the network believe the attacker is a neighbor. Hence, the flooder is identified from a greater distance. In the case of the sinkhole attack, the node identifying the sinkhole will be a true neighbor of the sinkhole.

4.6.3.1 Sinkhole

Messages related to the identification process are generated and transmitted in the network only when an attacker is detected. As mentioned before, there are no additional network messages required for the detection process. The message overhead is dependent upon how far, or the number of hops h, the detecting node is from the attacker. Therefore, m_{Low} refers to the number of messages required for the assignments of L_{Low} along the path to the attacker. Also partially dependent upon the number of hops to the attacker are the number of neighbor list messages (m_{nl}) which are sent by nodes that have both the assigning node and L_{Low} in common. Whether the attacker participates in the process by assigning a L_{Low} greatly affects the number of messages generated. We therefore apply binomials j and k to reflect participation where $j = 1$ for no participation, and $k = 1$ if there is participation. The additional messages generated near the attacker when the sinkhole or HELLO Flooder fails to participate is m_f, with m_p reflecting the number of messages generated during the "splash back" beyond the attacker due to its participation. Finally, we include the messages generated by the average number of "voters", m_v . The number of messages required by the identification method can be approximated by the following equation:

$$m = m_{Low} + m_{nl} + km_p + jm_f + m_v. \tag{4.13}$$

This equation can be simplified by including the parameters for the number of hops and the average number of neighbors common to L_{n-1} and L_n in each hop. Here we use the grid network with an average of eight neighbors per node:

$$m = h + 2h + 5k + 3j + 3 \tag{4.14}$$

or

$$m = 3h + 8k + 6j. \tag{4.15}$$

Table 4.15 provides the number of messages generated for the identification of the sinkhole by each individual alerted node during simulation using the original thirty node grid network with one compliant sinkhole. Note that "hops" is often equal to the "ring" as listed in the table. However, there are situations due to node placement on the ring in which two hops can occur on a ring. Figure **??** provides a graph depicting the actual messages sent by nodes on each ring compared to the estimated number of messages for the ring.

A protocol for sharing the burden of starting the sinkhole identification process would need to be developed in a real network, and is not in the scope of this paper. With the proper protocol, only one node would start the identification process, limiting the number of messages as overhead. Note that the "path" provided in the Table 4.15 is not the route used to send traffic; it is the path followed from the detecting node to the sinkhole through the nodes with the lowest neighbor reliability values.

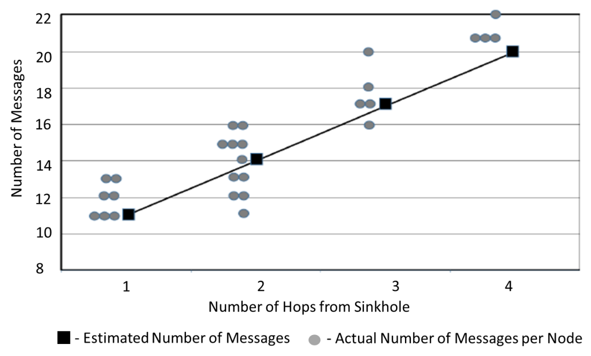

Figure 4.7 Number of Messages Per Ring (One Sinkhole)

4.6.3.2 HELLO Flood

Messages related to the identification process are generated and transmitted in the network only when an attacker is detected. As mentioned before, there are no additional network messages required for the detection process. The message overhead is dependent upon the number of hops h the detecting node is from the attacker itself, or from a node that believes the attacker is a neighbor. We include the messages generated by the average number of "voters", m_v for both voting and providing neighbor lists m_{nl}. Note that as we get close enough to the HELLO flooder *all* of the L_x neighbors believe they are neighbors to both L_{LOW} (the attacker) and L, so they all provide neighbor lists. We also include one message for the second query made by the assigning node that gets no reply from the attacker. The number of messages required by the identification method can be approximated by the following equation:

$$m = h + m_{nl} + m_v + 1. \tag{4.16}$$

Figure 4.8 provides a graph depicting the messages sent during simulation by nodes on each ring compared to the number of messages estimated by Equation 4.16 for the ring. As mentioned in the sinkhole message analysis, a protocol for sharing the burden of starting the HELLO flood attacker identification process would need to be developed in a real network, and is not in the scope of this paper. With the proper protocol, only one node would start the identification process, limiting the number of messages as overhead.

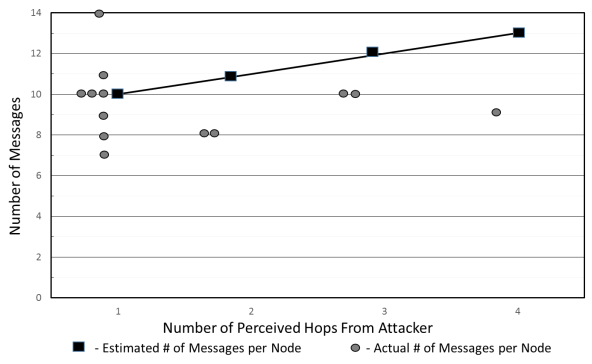

Figure 4.8 Number of Messages Per Ring (HELLO Flood Attack)

4.6.4 Steps Required for Attacker Detection and Identification

There are two phases that must be considered when modeling the number of required steps c for the described method. We assign c_d to the number of steps required in the detection phase, and c_{id} to the number of steps required for the identification phase. The following equation therefore reflects the total number of steps required in order to identify an attacker:

$$c = c_d + c_{id}. \tag{4.17}$$

In the detection phase a node assigned by an established burden-sharing protocol will use the stored packets received and sent data to calculate the neighbor reliabilities for its neighbors.

The node will then calculate the threshold, and compare the neighbor reliabilities to the the threshold. With c_{nr} representing the neighbor reliability calculation step, c_{thr} representing the threshold calculation, and c_{com} representing the comparison step, the steps to detection can be represented by the equation:

$$c_d = c_{nr} + c_{thr} + c_{com}. \tag{4.18}$$

Each of the factors in the equation for c_d is equal to one, so we can simplify this equation to $c_d = 3$.

The number of steps required for attacker identification is highly dependent upon how far, or the number of hops h, the detecting node is from the attacker. In the case of the HELLO Flood attacker, h represents the number of hops to a node that *perceives* itself as a neighbor to the attacker. This number is represented by c_h. We remind you that there are situations due to node placement on the ring in which two hops, or steps, can occur on a ring. There are additional steps depending upon whether the attacker participates in the identification process. To model this we again apply binomials j and k to reflect participation, where $j = 1$ for no participation and $k = 1$ if there is participation. In the case of the HELLO Flood attacker, we can expect $j = 1$ in most cases since the attacker will be too far from the identifiers to hear the identification and voting messages. Finally, we add the voting step, c_v which takes place at the end of the process. The number of steps required by the identification phase can therefore be represented by the equation:

$$c_{id} = c_h + jc_f + kc_p + c_v. \tag{4.19}$$

By including the parameters for the number of hops and the known subset number of steps for the participation and non-participation cases, the equation for the steps for identification can be simplified to:

$$c_{id} = h + j + 2k + 1 \tag{4.20}$$

or

$$c_{id} = h + 2j + 3k. \tag{4.21}$$

The total number of steps required for the detection and identification of an attacker can now be approximated by the equation.

$$c = 3 + h + 2j + 3k \tag{4.22}$$

Further simplification gives us the equation:

$$c = h + 5j + 3k. \tag{4.23}$$

The number of steps per route required by the identification process for the single sinkhole are listed in Table 4.15, and the number of steps required for HELLO flood attacker identification are listed in 4.16. The sinkhole analysis is for the simulation using the original thirty node grid network with one compliant sinkhole. We assume the attacker is participating in the process during simulation (if it is in range of message reception) when calculating the number of steps listed in the tables. Figure **??** provides a graph depicting the actual number steps required for the nodes in each ring to identify the sinkhole, as compared to the estimated number of steps. Figure 4.10 provides the same data for the HELLO flood attacker.

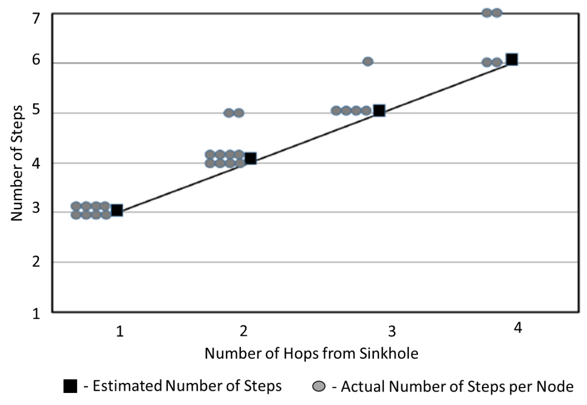

Figure 4.9 Number of Steps Per Ring (One Sinkhole)

4.7 Conclusion

In this paper we presented a hypothesis that, by adapting a methodology borrowed from the science of meteorology, we can utilize the data available at both the node and cooperative

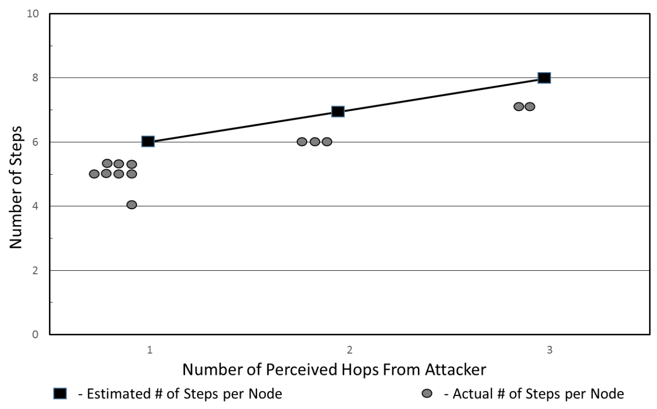

Figure 4.10 Number of Steps Per Ring (HELLO Flood Attack)

network levels to create a synoptic picture of the network health, providing indications of any intrusions or other network issues. Our major contribution is to provide a revolutionary way to analyze node and network data for patterns, dependence, and effects that indicate network issues at a distance. We collect node and network data, combine and manipulate it, and tease out information about the state of the network. By using the data gathered, such as packet delivery ratio, node reliability, and route reliability, we showed we could combine, analyze, and interpret the data to build a picture of the network state. We demonstrated this by simulating one or more sinkholes in a grid network of thirty nodes, one sinkhole in a scrambled network of thirty nodes, and a HELLO flood attacker in a scrambled network of twenty nodes. With proper mathematical analysis of the data we were able to build a network picture and identify the attackers with little energy expended for calculation or messages. This method did not rely upon the conventional methods of stored patterns for comparison, but only proper analysis of a subset of the real time parameters of the network.

Simulations using the network described in Table 4.12 showed that the original scheme found false sinkholes in networks with eight or fewer sinkhole nodes. However, the addition of a second threshold resulted in the elimination of the false positives, with the trade off of increased false negatives in networks of six or more sinkholes. This tradeoff is justified because we can realistically expect a network to have fewer than six (sinkhole) attackers. Therefore, using the two thresholds provides proper intrusion detection for this type of attack.

Table 4.15 Messages Generated in Identification Process (Sinkhole)

Ring (n)	Detecting Node	Path	Actual # of Steps	Actual # of Messages	Ring (n)	Detecting Node	Path	Actual # of Steps	Actual # of Messages
1	Q	Q-W	4	12	2	T	T-X-W	5	12
1	R	R-W	4	11	2	U	U-Z-V-W	6	16
1	S	S-W	4	11	2	Y	Y-CC-W	5	11
1	V	V-W	4	13	2	Z	Z-V-W	5	13
1	X	X-W	4	10	2	DD	DD-X-W	5	12
1	AA	AA-W	4	12	3	F	F-L-R-W	6	18
1	BB	BB-W	4	13	3	G	G-H-M-R-W	7	20
1	CC	CC-W	4	11	3	H	H-M-R-W	6	17
2	K	K-Q-W	5	15	3	I	I-N-R-W	6	17
2	L	L-R-W	5	15	3	J	J-N-R-W	6	16
2	M	M-R-W	5	16	4	A	A-G-H-M-R-W	8	21
2	N	N-R-W	5	13	4	B	B-H-M-R-W	7	21
2	O	O-T-X-W	6	14	4	C	C-G-H-M-R-W	8	22
2	P	P-V-W	5	15	4	D	D-H-M-R-W	7	21

Simulations in a larger network indicated the proposed method required an adjusted criteria to eliminate a number of the false positives. This result implies that the method described will work for differently sized networks. However, additional work needs to be done to determine standard criteria, or a method of determining network-based criteria, that is applicable to different sizes of networks. Additionally, Shannon's entropy equation was used for the analysis of the network results. It is possible that another entropy equation may provide finer results that may be less affected by the level of fidelity required by each network.

Our results included the numbers of nodes in the simulated networks that would start the identification process and identify the attackers presented based upon the threshold criteria. For

Table 4.16 Messages Generated in Identification Process (HELLO Flood)

Actual Ring (n) / Perceived Ring	Detecting Node	Path	Actual Number of Steps	Actual Number of Messages	Actual Ring (n) / Perceived Ring	Detecting Node	Path	Actual Number of Steps	Actual Number of Messages
1/1	S	S-M	4	11	3/1	K	K-M	5	7
2/1	E	E-M	5	8	3/2	Q	Q-R-M	6	8
2/1	H	H-M	5	14	4/2	G	G-A-M	6	8
2/1	C	C-M	5	10	4/3	D	D-G-A-M	7	9
2/1	R	R-M	5	10	4/3	U	U-Q-R-M	7	9
3/1	A	A-M	5	9	4/4	N	N-U-Q-R-M	8	9
3/1	F	F-M	5	10					

the sinkhole or HELLO flood attacker identification process to be implemented in a real network, a protocol for sharing the responsibility of identifying the attacker would need to be developed. The objective of the protocol would be to allow identification of the attacker while limiting the number of nodes starting the identification process, and the number of messages flooding the network. Possible strategies are a round robin system based upon time, or assignment of responsibility to cluster heads.

The synoptic analysis technique presented in this paper is founded upon comparing the counts of events in or effects on the wireless network. Other attacks that are based upon causing or affecting countable events that trigger changes in network characteristics are candidates for synoptic analysis. Attacks at the physical, network, and data link layers such as jamming and wormhole assaults are likely contenders. Challenges to the use of the technique described include the development of protocols to identify the initiating node(s) and the development of a proper network reaction. An additional challenge is a method to provide the network nodes with more distributed network data without increasing the controlling network traffic.

CHAPTER 5. CONCLUSION

With our increasing usage of the air as a medium for connecting electronically with the world, the current spectrum defined for commercial and personal usage has become crowded. This trend is expected to continue. The cognitive radio network with software defined capabilities will open to users more spectrum frequencies, and hence, enhanced communication opportunities. However, the new technology also provides avenues for new attacks perpetrated by malicious or selfish users with the desire to inhibit communication, capture or change the message, or use the spectrum exclusively.

In the second chapter of this dissertation we explored the structures of malicious attacks on the cognitive radio network. We identified attacks from both the traditional cellular networks and the wireless sensor network arena that apply to the cognitive radio network. We also presented attack scenarios specific to the cognitive radio network architecture and capabilities. Following each attack scenario we presented mitigating techniques particular to the attack.

In the third chapter we presented a hypothesis that, by adapting a methodology borrowed from the science of meteorology, we could utilize the data available at both the node and cooperative network levels to create a synoptic picture of the network health, providing indications of any intrusions or other network issues. Our major contribution was to provide a revolutionary way to analyze node and network data for patterns, dependence, and effects that indicate network issues at a distance. By using the data gathered, such as packet delivery ratio, node reliability, and route reliability, we showed we could combine, analyze, and interpret the data to build a picture of the network state. We demonstrated this by simulating one or more sinkholes in a grid network of thirty nodes. We expanded upon the premise with additional studies in the fourth chapter. We simulated one sinkhole in a scrambled network of thirty nodes, one sinkhole in hundred node grid network, and a HELLO flood attacker in a scrambled network of twenty nodes. With proper mathematical analysis of the data we were able to build a network picture and identify the attackers with little energy expended for calculation or messages. This method did not rely upon the conventional methods of stored patterns for comparison, but only proper analysis of a subset of the real time parameters of the network.

Simulations using the network described in Table 4.12 showed that the original scheme found false sinkholes in networks with eight or fewer sinkhole nodes. However, the addition of a second threshold resulted in the elimination of the false positives, with the trade off of increased false negatives in networks of six or more sinkholes. This tradeoff is justified because we can realistically expect a network to have fewer than six (sinkhole) attackers. Analysis of the data derived from the larger, one hundred node network indicated that new thresholding criteria was required for the larger network in order to eliminate false positives. Therefore, using the two thresholds provides proper intrusion detection for this type of attack, and the proper thresholding scheme is dependent upon the size of the network.

Our results included the numbers of nodes in the simulated networks that would start the identification process and identify the attackers presented based upon the threshold criteria. For the sinkhole or HELLO flood attacker identification process to be implemented in a real network, a protocol for sharing the responsibility of identifying the attacker would need to be developed. The objective of the protocol would be to allow identification of the attacker while limiting the number of nodes starting the identification process, and the number of messages flooding the network. Possible strategies are a round robin system based upon time, or assignment of responsibility to cluster heads.

The synoptic analysis technique presented in this dissertation is founded upon comparing the counts of events in or effects on the wireless network. Other attacks that are based upon causing or affecting countable events that trigger changes in network characteristics are candidates for synoptic analysis. Attacks at the physical, network, and data link layers such as jamming and wormhole assaults are likely contenders. Challenges to the use of the technique described include the development of protocols to identify the initiating node(s) and the development of a proper network reaction. An additional challenge is a method to provide the network nodes with more distributed network data without increasing the controlling network traffic.

The greatest advantage to incorporating the presented method into the intrusion detection toolbox is that the data required for analysis is all derived from the current network. Proper analysis of the data provides a window into the state of the current activity of the network, or its “network health”. By using carefully selected data we may be able to reduce the data overload experienced by the system administator, and instead identify and stop attacks on the network in a timely manner.

BIBLIOGRAPHY

[1] (2013 (Accessed: 27 May 2013)). What is unlicensed spectrum? what frequencies are they in? (wimax). [Online] http://www.wimax.com/wimax-regulatory/what-is-unlicensed-spectrum-what-frequencies-are-they-in. 5

[2] Abadie, A. and Wijesekera, D. (2012). Cognitive radio technologies: Envisioning the realization of network-centric warfare. In *Communications and Information Systems Conference (MilCIS), 2012 Military*, pages 1–7. IEEE. 7

[3] Abduvaliyev, A., Lee, S., and Lee, Y.-K. (2010). Energy efficient hybrid intrusion detection system for wireless sensor networks. In *Electronics and Information Engineering (ICEIE), 2010 International Conference On*, volume 2, pages V2–25. IEEE. 58, 87

[4] Arshad, K. (2012). Malicious users detection in collaborative spectrum sensing using statistical tests. In *Ubiquitous and Future Networks (ICUFN), 2012 Fourth International Conference on*, pages 109–113. IEEE. 26, 30

[5] Attar, A., Tang, H., Vasilakos, A. V., Yu, F. R., and Leung, V. C. (2012). A survey of security challenges in cognitive radio networks: Solutions and future research directions. *Proceedings of the IEEE*, (Volume:100 , Issue: 12):3172 – 3186. 45

[6] Awerbuch, B., Curtmola, R., Holmer, D., Nita-Rotaru, C., and Rubens, H. (2008). Odsbr: An on-demand secure byzantine resilient routing protocol for wireless ad hoc networks. *ACM Transactions on Information and System Security (TISSEC)*, 10(4):6. 59, 88

[7] Bahrak, B. and Park, J.-M. (2013). Security of spectrum learning in cognitive radios. *arXiv preprint arXiv:1304.0606*. 9, 17

[8] Baldini, G., Rakovic, V., Atanasovski, V., and Gavrilovska, L. (2012a). Security aspects of policy controlled cognitive radio. In *New Technologies, Mobility and Security (NTMS), 2012 5th International Conference on*, pages 1–5. IEEE. 6, 44, 46, 52

[9] Baldini, G., Sturman, T., Biswas, A. R., Leschhorn, R., Godor, G., and Street, M. (2012b). Security aspects in software defined radio and cognitive radio networks: a survey and a way ahead. *Communications Surveys & Tutorials, IEEE*, 14(2):355–379. 6

[10] Bian, K., Du, X., and Li, X. (2013). Enabling fair spectrum sharing: Mitigating selfish misbehaviors in spectrum contention. *IEEE Network*, page 17. 31

[11] Blasch, E., Busch, T., Kumar, S., and Pham, K. (2013). Trends in survivable/secure cognitive networks. In *Computing, Networking and Communications (ICNC), 2013 International Conference on*, pages 825–829. IEEE. 6

[12] Camilo, M., Moura, D., Galdino, J., and Salles, R. M. (2012). Anti-jamming defense mechanism in cognitive radios networks. In *MILITARY COMMUNICATIONS CONFERENCE, 2012-MILCOM 2012*, pages 1–6. IEEE. 9, 21

[13] Cao, Z., Zhou, X., Xu, M., Chen, Z., Hu, J., and Tang, L. (2006). Enhancing base station security against dos attacks in wireless sensor networks. In *Wireless Communications, Networking and Mobile Computing, 2006. WiCOM 2006. International Conference on*, pages 1–4. IEEE. 90

[14] Cassola, A., Jin, T., Noubir, G., and Thapa, B. (2013). Efficient spread spectrum communication without preshared secrets. *Mobile Computing, IEEE Transactions on*, 12(8):1669–1680. 9, 25

[15] Chan, A., Liu, X., Noubir, G., and Thapa, B. (2007). Broadcast control channel jamming: Resilience and identification of traitors. In *Information Theory, 2007. ISIT 2007. IEEE International Symposium on*, pages 2496–2500. IEEE. 26, 34

[16] Chen, C., Cheng, H., and Yao, Y.-D. (2011). Cooperative spectrum sensing in cognitive ra-

dio networks in the presence of the primary user emulation attack. *Wireless Communications, IEEE Transactions on*, 10(7):2135–2141. 9, 13

[17] Chen, C., Song, M., Xin, C., and Alam, M. (2012). A robust malicious user detection scheme in cooperative spectrum sensing. *Global Communications Conference (GLOBECOM), 2012 IEEE*, pages 4856 – 4861. 9, 16, 17

[18] Chen, C., Song, M., Xin, C., and Backens, J. (2013). A game-theoretical anti-jamming scheme for cognitive radio networks. *IEEE Network*. 23

[19] Chen, R., Park, J.-M., and Reed, J. H. (2008). Defense against primary user emulation attacks in cognitive radio networks. *Selected Areas in Communications, IEEE Journal on*, 26(1):25–37. 13

[20] Chin, W.-L., Tseng, C.-L., Tsai, C.-S., Kao, W.-C., and Kao, C.-W. (2012). Channel-based detection of primary user emulation attacks in cognitive radios. In *Vehicular Technology Conference (VTC Spring), 2012 IEEE 75th*, pages 1–5. IEEE. 9, 13, 14

[21] Clancy, T. C. and Goergen, N. (2008). Security in cognitive radio networks: Threats and mitigation. In *Cognitive Radio Oriented Wireless Networks and Communications, 2008. CrownCom 2008. 3rd International Conference on*, pages 1–8. IEEE. 9, 16, 44, 46

[22] Culpepper, B. J. and Tseng, H. C. (2004). Sinkhole intrusion indicators in dsr manets. In *Broadband Networks, 2004. BroadNets 2004. Proceedings. First International Conference on*, pages 681–688. IEEE. 60, 89

[23] Di Pietro, R. and Oligeri, G. (2013). Jamming mitigation in cognitive radio networks. *IEEE Network*. 23

[24] Douceur, J. R. (2002). The sybil attack. In *Peer-to-peer Systems*, pages 251–260. Springer. 36, 41

[25] Du, D. (2012). Soft reputation-based secure cooperative spectrum sensing. In *Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference on*, volume 1, pages 463–467. IEEE. 26, 28

[26] Duan, L., Min, A. W., Huang, J., and Shin, K. G. (2012). Attack prevention for collaborative spectrum sensing in cognitive radio networks. *Selected Areas in Communications, IEEE Journal on*, 30(9):1658–1665. 26, 31

[27] Dubey, R., Sharma, S., and Chouhan, L. (2012). Secure and trusted algorithm for cognitive radio network. In *Wireless and Optical Communications Networks (WOCN), 2012 Ninth International Conference on*, pages 1–7. IEEE. 9, 11, 12, 26

[28] Ehsan, H. and Khan, F. A. (2012). Malicious aodv: Implementation and analysis of routing attacks in manets. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*, pages 1181–1187. IEEE. 68, 99, 110

[29] El-Hajj, W., Safa, H., and Guizani, M. (2011). Survey of security issues in cognitive radio networks. *Journal of Internet Technology*, 12(2):181–198. 36, 41, 47, 50

[30] Farrag, M., El-Khamy, M., and El-Sharkawy, M. (2012). C44. secure cooperative blindly-optimized compressive spectrum sensing for cognitive radio. In *Radio Science Conference (NRSC), 2012 29th National*, pages 533–540. IEEE. 26, 28

[31] Firouzbakht, K., Noubir, G., and Salehi, M. (2012). On the capacity of rate-adaptive packetized wireless communication links under jamming. In *Proceedings of the fifth ACM conference on Security and Privacy in Wireless and Mobile Networks*, pages 3–14. ACM. 9, 23

[32] Fragkiadakis, A., Tragos, E., and Askoxylakis, I. (2013). A survey on security threats and detection techniques in cognitive radio networks. *Communications Surveys & Tutorials, IEEE*, (Volume:15 , Issue: 1):428–445. 6

[33] Gandhewar, N. and Patel, R. (2012). Detection and prevention of sinkhole attack on aodv protocol in mobile adhoc network. In *Computational Intelligence and Communication Networks (CICN), 2012 Fourth International Conference on*, pages 714–718. IEEE. 60, 89

[34] Gerrigagoitia, K., Uribeetxeberria, R., Zurutuza, U., and Arenaza, I. (2012). Reputation-based intrusion detection system for wireless sensor networks. In *Complexity in Engineering (COMPENG), 2012*, pages 1–5. IEEE. 58, 87

[35] Haghighat, M. and Sadough, S. M.-S. (2012). Cooperative spectrum sensing in cognitive radio networks under primary user emulation attacks. In *Telecommunications (IST), 2012 Sixth International Symposium on*, pages 148–151. IEEE. 9, 13

[36] Hao, D. and Sakurai, K. (2012). A differential game approach to mitigating primary user emulation attacks in cognitive radio networks. In *Advanced Information Networking and Applications (AINA), 2012 IEEE 26th International Conference on*, pages 495–502. IEEE. 9, 11

[37] Harmer, P. K. and Temple, M. A. (2013). An improved lfs engine for physical layer security augmentation in cognitive networks. In *Computing, Networking and Communications (ICNC), 2013 International Conference on*, pages 719–723. IEEE. 9, 13

[38] He, X., Dai, H., and Ning, P. (May 2013). A byzantine attack defender in cognitive radio networks: The conditional frequency check. *Wireless Communications, IEEE Transactions*, (Volume:12 , Issue: 5):2512 – 2523. 26, 30

[39] Hernandez-Orallo, E., Serrat, M. D., Cano, J., Calafate, C. T., and Manzoni, P. (2012). Improving selfish node detection in manets using a collaborative watchdog. *Communications Letters, IEEE*, 16(5):642–645. 59, 89

[40] Hlavacek, D. and Chang, J. M. (2014). A layered approach to cognitive radio network security: A survey. *Computer Networks*, 75:414–436. 2, 90

[41] Hlavacek, D. T. and Chang, J. M. (2015). Design and analysis of a method for synoptic level network intrusion detection. In *2015 IEEE 39th Annual International Computers, Software & Applications Conference (accepted)*, Taichung, Taiwan. 3, 85

[42] Hou, L., Yeung, K., and Wong, K. (2012). A virus spreading model for cognitive radio networks. *Physica A: Statistical Mechanics and its Applications*. 44, 45

[43] Hu, N., Yao, Y.-D., and Mitola, J. (2012). Most active band (mab) attack and countermeasures in a cognitive radio network. *Wireless Communications, IEEE Transactions on*, 11(3):898–902. 22

[44] Hu, Y.-C., Perrig, A., and Johnson, D. B. (2003). Packet leashes: a defense against wormhole attacks in wireless networks. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 3, pages 1976–1986. IEEE. 36, 39

[45] Jana, S., Zeng, K., and Mohapatra, P. (2012). Trusted collaborative spectrum sensing for mobile cognitive radio networks. In *INFOCOM, 2012 Proceedings IEEE*, pages 2621–2625. IEEE. 26, 29

[46] Jebadurai, J. V. S., Melvin, A., and Jebadurai, I. (2011). Sinkhole detection in mobile ad-hoc networks using mutual understanding among nodes. In *Electronics Computer Technology (ICECT), 2011 3rd International Conference on*, volume 3, pages 321–324. IEEE. 60, 89

[47] Jiang, N., Hua, K. A., and Liu, D. (2007). A scalable and robust approach to collaboration enforcement in mobile ad-hoc networks. *Journal of Communications and Networks*, 9(1):56. 47, 51

[48] Jin, Z., Anand, S., and Subbalakshmi, K. (2012). Impact of primary user emulation attacks on dynamic spectrum access networks. 9, 10

[49] Jo, M., Han, L., Kim, D., and In, H. P. (2013). Selfish attacks and detection in cognitive radio ad-hoc networks. *IEEE Network*, 27(3). 27

[50] Kalamkar, S. S., Banerjee, A., and Roychowdhury, A. (2012). Malicious user suppression for cooperative spectrum sensing in cognitive radio networks using dixon's outlier detection method. In *Communications (NCC), 2012 National Conference on*, pages 1–5. IEEE. 26, 30

[51] Karlof, C. and Wagner, D. (2003). Secure routing in wireless sensor networks: Attacks and countermeasures. *Ad hoc networks*, 1(2):293–315. 36, 37, 39, 40, 41, 42, 90

[52] Khare, A., Saxena, M., Thakur, R. S., and Chourasia, K. (2013). Attacks & preventions of cognitive radio network-a survey. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, 2(3):pp–1002. 6

[53] Kumar, S. and Joshi, R. (2011). Design and implementation of ids using snort, entropy and alert ranking system. In *Signal Processing, Communication, Computing and Networking Technologies (ICSCCN), 2011 International Conference on*, pages 264–268. IEEE. 57, 87

[54] Lazos, L., Liu, S., and Krunz, M. (2009). Mitigating control-channel jamming attacks in multi-channel ad hoc networks. In *Proceedings of the second ACM conference on Wireless network security*, pages 169–180. ACM. 26, 33

[55] León, O., Hernandez-Serrano, J., and Soriano, M. (2009). A new cross-layer attack to tcp in cognitive radio networks. In *Cross Layer Design, 2009. IWCLD'09. Second International Workshop on*, pages 1–5. IEEE. 47, 50

[56] León, O., Hernández-Serrano, J., and Soriano, M. (2010). Securing cognitive radio networks. *International Journal of Communication Systems*, 23(5):633–652. 9, 16

[57] Li, X., Lu, R., Liang, X., and Shen, X. (2011). Side channel monitoring: packet drop attack detection in wireless ad hoc networks. In *Communications (ICC), 2011 IEEE International Conference on*, pages 1–5. IEEE. 59, 88

[58] Li, Y., Fang, B.-X., Chen, Y., and Guo, L. (2006). A lightweight intrusion detection model based on feature selection and maximum entropy model. In *Communication Technology, 2006. ICCT'06. International Conference on*, pages 1–4. IEEE. 57, 87

[59] Lin, F., Hu, Z., Hou, S., Yu, J., Zhang, C., Guo, N., Wicks, M., Qiu, R. C., and Currie, K. (2011). Cognitive radio network as wireless sensor network (ii): Security consideration. In *Aerospace and Electronics Conference (NAECON), Proceedings of the 2011 IEEE National*, pages 324–328. IEEE. 9, 11

[60] Liu, S., Lazos, L., and Krunz, M. (2011). Thwarting inside jamming attacks on wireless

broadcast communications. In *Proceedings of the fourth ACM conference on Wireless network security*, pages 29–40. ACM. 9

[61] Liu, S., Lazos, L., and Krunz, M. (2012a). Thwarting control-channel jamming attacks from inside jammers. *Mobile Computing, IEEE Transactions on*, 11(9):1545–1558. 25, 26, 33

[62] Liu, S., Zhu, H., Li, S., Li, X., Chen, C., and Guan, X. (2012b). An adaptive deviation-tolerant secure scheme for distributed cooperative spectrum sensing. *Global Communications Conference (GLOBECOM), 2012 IEEE*, pages 603 – 608. 26, 28

[63] Lo, B. F. (2011). A survey of common control channel design in cognitive radio networks. *Physical Communication*, 4(1):26–39. 26, 32

[64] Lo, B. F. and Akyildiz, I. F. (2012). Multiagent jamming-resilient control channel game for cognitive radio ad hoc networks. In *Communications (ICC), 2012 IEEE International Conference on*, pages 1821–1826. IEEE. 9, 26, 35

[65] Lu, K., Ke, H., Yang, J., and Zhang, L. (2012). Research of pue attack based on location. In *Signal Processing (ICSP), 2012 IEEE 11th International Conference on*, volume 2, pages 1345–1348. IEEE. 9, 11

[66] Ma, L., Shen, C.-C., and Ryu, B. (2007). Single-radio adaptive channel algorithm for spectrum agile wireless ad hoc networks. In *New Frontiers in Dynamic Spectrum Access Networks, 2007. DySPAN 2007. 2nd IEEE International Symposium on*, pages 547–558. IEEE. 26, 32, 33

[67] Mathur, C. N. and Subbalakshmi, K. (2007). Security issues in cognitive radio networks. *Cognitive networks: towards self-aware networks, July.* 44, 45, 47, 48, 51

[68] Meghanathan, N. (2013a). A comprehensive review and analysis of solutions for different layers of the tcp/ip layer stack and security issues for cognitive radio networks. *International Journal of Advancements in Technology*, 4(1):1–27. 6

[69] Meghanathan, N. (2013b). A survey on the communication protocols and security in cognitive radio networks. *International Journal of Communication Networks and Information Security (IJCNIS)*, 5(1). 6

[70] Mihovska, A., Prasad, R., Tragos, E. Z., and Angelakis, V. (2012). Design considerations for a cognitive radio trust and security framework. In *Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), 2012 IEEE 17th International Workshop on*, pages 156–158. IEEE. 26, 27

[71] Mitola III, J. and Maguire Jr, G. Q. (1999). Cognitive radio: making software radios more personal. *Personal Communications, IEEE*, 6(4):13–18. 7

[72] Newsome, J., Shi, E., Song, D., and Perrig, A. (2004). The sybil attack in sensor networks: analysis & defenses. In *Proceedings of the 3rd international symposium on Information processing in sensor networks*, pages 259–268. ACM. 36, 41, 42

[73] Nezhadal, S. M. M., Berangi, R., and Fathy, M. (2012). Common control channel saturation detection and enhancement in cognitive radio networks. *International Journal*, 3. 26, 32

[74] Noh, G., Lim, S., Lee, S., and Hong, D. (2012). Goodness-of-fit-based malicious user detection in cooperative spectrum sensing. In *Vehicular Technology Conference (VTC Fall), 2012 IEEE*, pages 1–5. IEEE. 26, 28

[75] Noubir, G., Rajaraman, R., Sheng, B., and Thapa, B. (2011). On the robustness of ieee 802.11 rate adaptation algorithms against smart jamming. In *Proceedings of the fourth ACM conference on Wireless network security*, pages 97–108. ACM. 9, 25

[76] Parvin, S. and Hussain, F. K. (2012). Trust-based security for community-based cognitive radio networks. In *Advanced Information Networking and Applications (AINA), 2012 IEEE 26th International Conference on*, pages 518–525. IEEE. 26, 28

[77] Parvin, S., Hussain, F. K., Hussain, O. K., Han, S., Tian, B., and Chang, E. (2012). Cognitive radio network security: A survey. *Journal of Network and Computer Applications*. 6, 7

[78] Patel, M. H. P. and Chaudhari, M. B. (2012). Survey: Impact of jellyfish on wireless ad-hoc network. *International Journal of Engineering*, 1(9). 47, 51

[79] Pei, Q., Li, L., Li, H., and Yuan, B. (2012a). Adaptive trust management mechanism for cognitive radio networks. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*, pages 826–831. IEEE. 26, 28, 29

[80] Pei, Q., Yuan, B., Li, L., and Li, H. (2012b). A sensing and etiquette reputation-based trust management for centralized cognitive radio networks. *Neurocomputing*. 26, 28, 29

[81] Penna, F., Sun, Y., Dolecek, L., and Cabric, D. (2012). Detecting and counteracting statistical attacks in cooperative spectrum sensing. *Signal Processing, IEEE Transactions on*, 60(4):1806–1822. 26, 30

[82] Pirzada, A. A. and McDonald, C. (2006). Trust establishment in pure ad-hoc networks. *Wireless Personal Communications*, 37(1-2):139–168. 47, 51

[83] Popescu, A. (2012). Cognitive radio networks. *Communications (COMM), 2012 9th International Conference*, pages 11 – 15. 6

[84] Purewal, S. (October 2011 (Accessed: 27 May 2013)). "wireless devices outnumber us population, survey says" (pcworld). [Online] http://www.pcworld.com/article/241826/. 5

[85] Qian, L., Li, X., and Wei, S. (Jan 2013). Cross-layer detection of stealthy jammers in multihop cognitive radio networks. *Computing, Networking and Communications (ICNC), 2013 International Conference*, pages 1026 – 1030. 9, 20

[86] Qusay, H. and MAHMOU, D. (2007). *Cognitive networks: Towards self-aware networks*. London: Wiley. 44

[87] Rawat, A. S., Anand, P., Chen, H., and Varshney, P. K. (2011). Collaborative spectrum sensing in the presence of byzantine attacks in cognitive radio networks. *Signal Processing, IEEE Transactions on*, 59(2):774–786. 26, 28

[88] Ren, K. and Wang, Q. (2013). Opportunistic spectrum access: from stochastic channels to non-stochastic channels. *Wireless Communications, IEEE*, 20(3). 6

[89] Samad, F. (2011). *Securing wireless mesh networks: a three dimensional perspective.* PhD thesis, Universitts bibliothek. 47, 51

[90] Sen, J. (2013). Security and privacy challenges in cognitive wireless sensor networks. *arXiv preprint arXiv:1302.2253.* 6

[91] Shah, M. A., Zhang, S., and Maple, C. (2012). A novel multi-fold security framework for cognitive radio wireless ad-hoc networks. In *Automation and Computing (ICAC), 2012 18th International Conference on*, pages 1–6. IEEE. 52

[92] Shu, Z., Qian, Y., and Ci, S. (2013). On physical layer security for cognitive radio networks. *IEEE Network*, 27(3). 6

[93] Singh, S. and Trivedi, A. (2012). Anti-jamming in cognitive radio networks using reinforcement learning algorithms. In *Wireless and Optical Communications Networks (WOCN), 2012 Ninth International Conference on*, pages 1–5. IEEE. 9, 22

[94] Singh, V. P., Ukey, A. S. A., and Jain, S. (2013). Signal strength based hello flood attack detection and prevention in wireless sensor networks. *International Journal of Computer Applications*, 62(15). 90

[95] Sodagari, S. and Clancy, T. C. (2011). An anti-jamming strategy for channel access in cognitive radio networks. In *Decision and Game Theory for Security*, pages 34–43. Springer. 9, 24

[96] Song, Y. and Xie, J. (2012). Finding out the liars: Fighting against false channel information exchange attacks in cognitive radio ad hoc networks. *Global Communications Conference (GLOBECOM), 2012 IEEE*, pages 2095 – 2100. 9, 17

[97] Sorrells, C., Qian, L., and Li, H. (2012). Quickest detection of denial-of-service attacks in cognitive wireless networks. In *Homeland Security (HST), 2012 IEEE Conference on Technologies for*, pages 580–584. IEEE. 9, 20

[98] Strasser, M., Capkun, S., and Cagalj, M. (2008). Jamming-resistant key establishment using uncoordinated frequency hopping. In *Security and Privacy, 2008. SP 2008. IEEE Symposium on*, pages 64–78. IEEE. 9, 21, 24

[99] Sumathi, A. and Vidhyapriya, R. (2012). Security in cognitive radio networks-a survey. In *Intelligent Systems Design and Applications (ISDA), 2012 12th International Conference on*, pages 114–118. IEEE. 6

[100] Sutton, R. S. and Barto, A. G. (1998). *Reinforcement learning: An introduction*, volume 1. Cambridge Univ Press. 17

[101] Tague, P., Li, M., and Poovendran, R. (2007). Probabilistic mitigation of control channel jamming via random key distribution. In *Personal, Indoor and Mobile Radio Communications, 2007. PIMRC 2007. IEEE 18th International Symposium on*, pages 1–5. IEEE. 26, 34

[102] Tague, P., Li, M., and Poovendran, R. (2009). Mitigation of control channel jamming under node capture attacks. *IEEE Transactions on Mobile Computing*, 8(9). 26, 34

[103] Tan, K., Jana, S., Pathak, P. H., and Mohapatra, P. (2013). On insider misbehavior detection in cognitive radio networks. *IEEE Network*, page 5. 26

[104] Tan, Y., Hong, K., Sengupta, S., and Subbalakshmi, K. (2011). Using sybil identities for primary user emulation and byzantine attacks in dsa networks. In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pages 1–5. IEEE. 41

[105] Tan, Y., Sengupta, S., and Subbalakshmi, K. (2012). Primary user emulation attack in dynamic spectrum access networks: a game-theoretic approach. *Communications, IET*, 6(8):964–973. 36

[106] Tang, L. and Wu, J. (2012). Research and analysis on cognitive radio network security. *Wireless Sensor Network*, 4(4):120–126. 6

[107] Times, N. Y. (2014 (Accessed: 9 December 2014)). Cybersecurity requires more than 'patch and pray. [Online] http://www.sfgate.com/business/article/Cybersecurity-requires-more-than-patch-and-5938625.php. 1, 2, 83, 84

[108] Umang, S., Reddy, B., and Hoda, M. (2010). Enhanced intrusion detection system for malicious node detection in ad hoc routing protocols using minimal energy consumption. *IET communications*, 4(17):2084–2094. 58, 87

[109] Vernekar, D. S. (2012). An investigation of security challenges in cognitive radio networks. *http://digitalcommons.unl.edu/ceendiss/20/*. 6

[110] Vigna, G., Gwalani, S., Srinivasan, K., Belding-Royer, E. M., and Kemmerer, R. A. (2004). An intrusion detection tool for aodv-based ad hoc wireless networks. In *Computer Security Applications Conference, 2004. 20th Annual*, pages 16–27. IEEE. 57, 86

[111] Wang, B., Wu, Y., Liu, K. R., and Clancy, T. C. (2011a). An anti-jamming stochastic game for cognitive radio networks. *Selected Areas in Communications, IEEE Journal on*, 29(4):877–889. 9, 22

[112] Wang, Q., Ren, K., and Ning, P. (2011b). Anti-jamming communication in cognitive radio networks with unknown channel statistics. In *Network Protocols (ICNP), 2011 19th IEEE International Conference on*, pages 393–402. IEEE. 24

[113] Wang, Q., Xu, P., Ren, K., and Li, X.-Y. (2012a). Towards optimal adaptive ufh-based anti-jamming wireless communication. *Selected Areas in Communications, IEEE Journal on*, 30(1):16–30. 24

[114] Wang, W., Bhattacharjee, S., Chatterjee, M., and Kwiat, K. (2012b). Collaborative jamming and collaborative defense in cognitive radio networks. *Pervasive and Mobile Computing*. 9, 21

[115] Wang, W., Li, H., Sun, Y., and Han, Z. (2009). Attack-proof collaborative spectrum sensing in cognitive radio networks. In *Information Sciences and Systems, 2009. CISS 2009. 43rd Annual Conference on*, pages 130–134. IEEE. 9, 19

[116] Wang, W., Sun, Y., Li, H., and Han, Z. (2010). Cross-layer attack and defense in cognitive radio networks. In *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, pages 1–6. IEEE. 47, 49

[117] Wen, H., Li, S., Zhu, X., and Zhou, L. (2013). A framework of the phy-layer approach to defense against security threats in cognitive radio networks. *IEEE Network*. 13

[118] Wu, B., Chen, J., Wu, J., and Cardei, M. (2007). A survey of attacks and countermeasures in mobile ad hoc networks. In *Wireless Network Security*, pages 103–135. Springer. 36, 38

[119] Wu, Y., Wang, B., Liu, K., and Clancy, T. (2012). Anti-jamming games in multi-channel cognitive radio networks. *Selected Areas in Communications, IEEE Journal on*, 30(1):4–15. 9, 23

[120] Wyglinski, A. M., Nekovee, M., and Hou, T. (2009). *Cognitive radio communications and networks: principles and practice*. Academic Press. 9, 10

[121] Xiao, H., Yang, K., Wang, X., and Shao, H. (2012a). A robust mdp approach to secure power control in cognitive radio networks. In *Communications (ICC), 2012 IEEE International Conference on*, pages 4642–4647. IEEE. 9, 17

[122] Xiao, L., Lin, W. S., Chen, Y., and Liu, K. (2012b). Indirect reciprocity game modelling for secure wireless networks. In *Communications (ICC), 2012 IEEE International Conference on*, pages 928–933. IEEE. 28, 36, 42

[123] Yadav, S. and Nene, M. J. (2013). Rss based detection and expulsion of malicious users from cooperative sensing in cognitive radios. In *Advance Computing Conference (IACC), 2013 IEEE 3rd International*, pages 181–184. IEEE. 26, 31

[124] Yan, Q., Li, M., Jiang, T., Lou, W., and Hou, Y. T. (2012). Vulnerability and protection for distributed consensus-based spectrum sensing in cognitive radio networks. In *INFOCOM, 2012 Proceedings IEEE*, pages 900–908. IEEE. 9, 16

[125] Yi, S., Naldurg, P., and Kravets, R. (2001). Security-aware ad hoc routing for wireless networks. In *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, pages 299–302. ACM. 36, 38

[126] Yuan, Z., Niyato, D., Li, H., Song, J. B., and Han, Z. (2012). Defeating primary user emulation attacks using belief propagation in cognitive radio networks. *Selected Areas in Communications, IEEE Journal on*, 30(10):1850–1860. 9, 11, 12

[127] Zargar, S. T., Weiss, M. B., Caicedo, C. E., and Joshi, J. B. (2011). Security in dynamic spectrum access systems: A survey. *University of Pittsburgh http://d-scholarship.pitt.edu/2823/*. 9, 18

[128] Zhang, C., Yu, R., and Zhang, Y. (2012a). Performance analysis of primary user emulation attack in cognitive radio networks. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2012 8th International*, pages 371–376. IEEE. 9, 14

[129] Zhang, H., Liu, Z., and Hui, Q. (2012b). Optimal defense synthesis for jamming attacks in cognitive radio networks via swarm optimization. In *Computational Intelligence for Security and Defence Applications (CISDA), 2012 IEEE Symposium on*, pages 1–8. IEEE. 9, 22

[130] Zhang, L., Pei, Q., and Li, H. (2012c). Anti-jamming scheme based on zero pre-shared secret in cognitive radio network. In *Computational Intelligence and Security (CIS), 2012 Eighth International Conference on*, pages 670–673. IEEE. 26, 34

[131] Zhang, Y. and Lazos, L. (2013). Vulnerabilities of cognitive radio mac protocols and countermeasures. *IEEE Network*, page 41. 47, 49

[132] Zhao, C., Wang, W., Huang, L., and Yao, Y. (2009). Anti-pue attack base on the transmitter fingerprint identification in cognitive radio. In *Wireless Communications, Networking and Mobile Computing, 2009. WiCom'09. 5th International Conference on*, pages 1–5. IEEE. 9, 13, 14

[133] Zhao, J. and Cao, G. (2012). Robust topology control in multi-hop cognitive radio networks. In *INFOCOM, 2012 Proceedings IEEE*, pages 2032–2040. IEEE. 36, 43

[134] Zheng, X., Li, Y., and Zhang, H. (2010). A collision-free resident channel selection based solution for deafness problem in the cognitive radio networks. In *Wireless Information Technology and Systems (ICWITS), 2010 IEEE International Conference on*, pages 1–4. IEEE. 47, 48

[135] Zhijie, H. and Ruchuang, W. (2012). Intrusion detection for wireless sensor network based on traffic prediction model. *Physics Procedia*, 25:2072–2080. 58, 87