Performance Analysis of CR-honeynet to prevent Jamming Attack through Stochastic Modeling ☆

Suman Bhunia[a], Shamik Sengupta[a], Felisa Vázquez-Abad[b]

[a] *Dept. of Computer Science and Engineering, University of Nevada, Reno, USA 89557*
[b] *Dept. of Computer Science, Hunter College, City University of New York, USA 10065*

Abstract

Cognitive Radio Network (CRN) has to stall its packet transmission periodically to sense the spectrum for Primary User's (PU's) transmission. The limited and dynamically available spectrum and fixed periodic schedule of transmission interruption makes it harder to model the performance of a CRNs. Again, an open and dynamic spectrum access model brings forth a serious challenge of sustenance among the CRN and makes them more susceptible to jamming-based denial of service (DoS) attacks. Inspired by honeypot in the network security, we propose a honeynet based defense mechanism called CR-honeynet. CR-honeynet aims to avoid attacks on legitimate communications by dedicating a Secondary User (SU) as a honeynode, to deter the attacker from attacking legitimate SUs and attack the honeynode instead. Dedicating an SU as honeynode, on account of its permanent idleness, is wasteful of an entire node as a resource. We seek to resolve the dilemma by dynamically selecting the honeynode for each transmission period. The contribution of the current paper is two-fold. Initially, we develop the first comprehensive queuing model for CRNs, which pose unique modeling challenges, due to their fixed periodic sensing and transmission cycles. In the second step, we introduce a series of strategies for honeynode selection to combat these attacks while keeping the CRN's performance optimal for different traffic scenarios. We build a simulation of a CRN under jamming attack and analyze its performance with different honeynode selection strategies. We find that the predictions, of our mathematical model, track closely with the results of our simulation experiments.

Keywords: Cognitive Radio, Jamming attack, Honeynet, Stochastic Model, Queuing theory, queue with vacation

☆ This research was supported by NSF CAREER grant CNS #1346600.

Email addresses: sumanbhunia@gmail.com (Suman Bhunia), felisav@hunter.cuny.edu (Felisa Vázquez-Abad)

URL: http://www.cse.unr.edu/~shamik (Shamik Sengupta)

1. Introduction

The conventional static spectrum allocation policy has resulted in suboptimal use of spectrum resource, leading to over-utilization in some bands and under-utilization in others [1]. As a solution, dynamic spectrum access-based Cognitive Radio (CR) has been proposed. CR allows secondary users (SUs) to use an idle licensed spectrum while the proprietary primary user (PU) is not transmitting. The IEEE 802.22 [2] which is an emerging standard for CR-based wireless regional area networks (WRANs), aims at a vacant licensed TV spectrum to be used by SU without causing interference to PU. Infrastructure-based cognitive radio networks (CRNs) consist of two major components: a central controller (such as base station or access point) and mobile SUs. The central controller supervises the communication and makes the spectrum allocation decisions. A sample CRN is presented in Fig. 1.

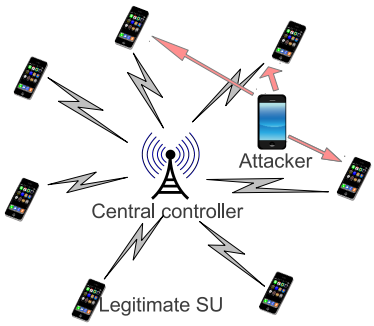


Figure 1: A sample CRN with an attacker

The dynamic nature of the available spectrum makes CRNs vulnerable to several spectrum etiquette attacks. The IEEE 802.22 standard does not specifically address the SU-SU interaction or SU protection, although it proactively specifies the PU protection. The "open" philosophy of the CR paradigm makes such networks susceptible to attacks by smart malicious users that could even render the legitimate CR spectrum-less [1, 3, 4]. Due to software reconfigurability, CRs can even be manipulated to disrupt other CRNs or legacy wireless networks with even greater impact than traditional hardware radios. The jamming-based Denial of Service (DoS) [1] attack is achieved by transmitting energy on the channel where a legitimate SU is communicating. An attacker can scan through channels, identify ongoing legitimate SU communication and then transmit a jamming signal on that particular channel causing heavy interference to the SU, which in effect, can block the legitimate SU's transmission completely.

A number of defense mechanisms against such attacks have been attempted [5–10]. Most of these techniques have considered that the attacker is naive and does not evolve. Inspired by "honeypot" in cybercrime, we propose *CR-honeynet*, which passively learns the attacker's strategy of assault and then dedicates an SU as an active decoy to lure the attacker to hit the decoy node. In this way, the assailant gets false satisfaction of attack, while legitimate SUs bypass attacks. In our earlier paper [11], We introduced the learning mechanism of CR-Honeynet. However the effectiveness of CR-Honeynet in CRN has to be studied before a CRN can deploy CR-Honeynet mechanism. The goal of this paper is to investigate whether allocating resources for CR-honeynet can be beneficial for improving system performance.

To protect PU incumbent services, DSA strictly enforces SUs to periodically pause its transmission and sense for PU activity. SUs scan the wireless envi-

Figure 2: Time domain representation of Cognitive Cycle

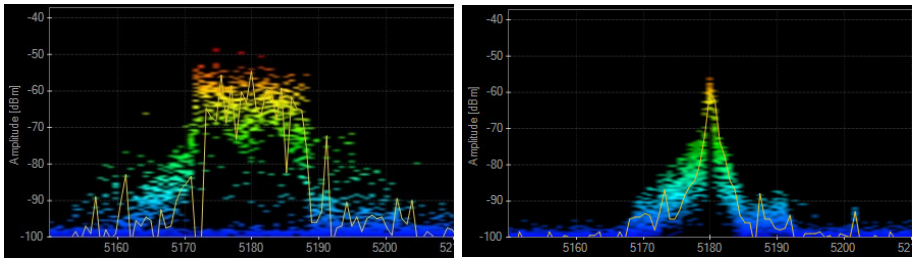
ronment for free channels in the *sensing period* and transmit packets during *transmission period*. This cognitive cycle is depicted in Fig. 2. The centralized controller allocates different channels to each SU. Several practical challenges need to be co-opted and addressed before allocating resource for honeynet in CRNs. Although dedicating an SU as honeynode potentially makes the CRN robust, it is not a "free ride" as it degrades the effective system throughput. Critical question is how would the honeynode be chosen then? Who will be responsible ("honeynode" selection) for auxiliary communications and monitoring in honeynode? To answer the above questions, we must first understand the complexity of the CRN's traffic behavior under DSA scenario. Consider a scenario wherein a user is conducting a number of simultaneous transmissions - for example, videoconferencing, and many more. All these applications generate packets randomly and independent of other applications. The complex nature of data traffic makes it difficult to analyze the Quality of Service (QoS). CRNs, meanwhile, exhibit a unique behavior pattern that remains yet to be investigated by any mathematical model. For example, the periodic sensing by SUs forces interruption on transmission, affecting end-to-end QoS by imposing delay and jitter on packet transmission. Thus, a major goal of this work is to model a CRN's service using stochastic analysis and use our model to estimate baseline performance indicators. Then we propose state dependent honeynode selection policies for different traffic models to enhance the CRN's performance.

The rest of the paper proceeds as follows: In Section 2, we discuss the motivation for our work, i.e., DoS attacks and honeynet limitations. Section 3 presents a mathematical model to estimate CRN performance using a queue with fixed periodic server vacation. Section 4 presents several honeynode selection policies. In section 5, we build a comprehensive simulator to study the performance of the proposed model, describe a utility model to determine when a honeynet can be used and when not, measure the fairness of all honeynode selection strategies and finally present the benefits of an optimal honeynode selection strategy that provides the best performance with fairness. Finally, section 6 concludes the paper.

2. Motivation

2.1. Jamming Attack

The threat of penalty can discourage a potential assailant from attacking a PU; however, when an SU accesses a channel, it borrows the channel, and it does not have any ground from which it can fend off attackers. While PUs are able to discourage attackers, SUs are left vulnerable to malicious jamming / disruptive attacks [12]. Jamming can be broadly categorized into two types [7, 10]. The first type being *physical layer jamming* where the attacker jams the channel of



(a) Normal communication (b) Jamming Signal

Figure 3: PSD for data communication and jamming signal

communication by sending strong noise or a jamming signal and the second type is *datalink / MAC layer jamming*, which targets several vulnerabilities present in the MAC layer protocol.

We run an experiment in our lab, to study the feasibility of jamming attack. Two computers are configured to communicate over a WLAN (IEEE 802.11-a, channel 36, central frequency: 5.18 GHz). The Power spectrul Density (PSD) for normal communication (without jamming), which can be seen in Fig. 3a is observed through Wi-spy spectrum analyzer [13]. Then we begin transmitting a narrow-band jamming signal of 2MHz from a GNU Radio [14] enabled USRP radio [15] on the same channel. In the presence of the jamming signal, the genuine transmission of the WLAN was stopped completely which can be observed in Fig. 3b. Here, the attacker is exploiting the vulnerability present in IEEE 802.11 MAC that enforces a node to sense the channel before transmission. When the legitimate transmitter senses that there is some energy on the channel, it refrains from transmission. Irrespective of the jamming technique, a target node suffers significant amount of data or packet loss and sometimes completely loses the channel. CRN, being a next generation intelligent network, should incorporate a mechanism to mitigate, avoid, and prevent such attacks.

2.2. Existing Jamming Detection and Prevention Mechanisms

Due to the noise in wireless medium, detection of jamming is challenging in combat with an attacker. A good survey of different detection mechanisms for the jamming-based DoS attack has been presented in [16]. It is difficult to correctly detect jamming, based on a single system parameter. Several system parameters such as received-signal-strength, packet-send-ratio, packet-delivery-ratio, carrier-sensing-time, etc., are used to model jamming detection systems. Consistency checks among system parameters are used for more efficient detection. Authors of [17] have classified spectrum usage anomaly detection data fusion algorithms. Through different fusion algorithms, anomalies in spectrum usage can be detected successfully with higher efficiency. A cross-layer detection mechanism of anomalous spectrum attack has been proposed in [18], where the network maps the jammed geographical region, using spectrum-sensing reports collected from each SUs that are equipped with localization modules.

Existing defense mechanisms can be broadly categorized into *Channel Surfing*, *Spatial Retreat*, *Mapping Jammed Region*, *Spread Spectrum*, etc. In *Channel*

Surfing technique, the node which is under attack, migrates its channel of communication upon detection of jamming [5]. Authors of [6] proposed proactive frequency hopping, where the nodes change its channel of communication, irrespective of attacks to avoid jamming. The authors considered a fixed number of channels that the attacker can use, that is known to an SU, which in reality is difficult to achieve. In *Spatial Retreat* [7], mobile nodes relocate themselves physically to avoid jamming. The constraint of this approach is that the nodes are required to be highly mobile, which is not applicable for static nodes. In *Mapping Jammed Region* [8] approach, the multi-hop, and intensely populated, CRN avoids routing through the links that have been affected by jamming. This mechanism fails if there is only one path and that path is attacked. Majority of current countermeasures defend against jamming after it has been detected; on the contrary, CR-Honeynet learns from the history of attack and provide proactive defense mechanism.

2.3. Use of Honeynet in avoiding attacks

"Honeypot," in cybercrime, is defined as "a security resource who's value lies in being probed, attacked or compromised". In cybercrime defense, honeypots are being used as a camouflaging security tool with little or no actual production value to lure the attacker into giving them a false sense of satisfaction, thus bypassing (reducing) the attack impact and giving the defender a chance to retrieve valuable information about the attacker and their activities. This node is called honeynode. A single channel honeypot-based channel surfing, to mitigate jamming-based DoS attacks, has been proposed in [10]. The network dedicates a node, as honeypot, to monitor attacks. Upon detection of attack, the network switches its channel of operation, which results in long-time communication disruption. Majority of the previous works have assumed that the attacker is naive and does not evolve. Thus, none of these works have focused on learning the strategy of attacker where the attacker is also dynamic and changes its strategy of choosing the target communication characteristics.

From an intelligent and rational attacker's perspective, jamming a communication randomly will not yield optimal results; rather, an attacker can be most disruptive if it targets the communication that impacts the CRN most severely upon interruption [16, 19–21]. The attacker succeeds in determining highest impacting communication by observing certain transmission characteristics, for example, highest transmission power, highest data rate, modulation scheme, packet inter arrival time, quality of route with end-to-end acknowledgments, etc. [16]. To proactively defend against such intelligent attackers, a CRN must learn about the strategy that the attacker uses, to figure out the highest impacting communication. The attacker's strategy of finding the highest impacting communication can be used as a trap by the defending CRN to detract the attacker from striking legitimate communications.

We propose *CR-honeynet*, a honeynet-based defense mechanism where the CRN passively learns the *strategy of the attacker* and then places an active decoy, namely *honeynode* to entice the attacker for jamming the honeynode transmission. Thus, the attacker gets a false impression of attacking the highest

impacting communication, whereas legitimate SU communications avoid attacks and reduce attack impact on the CRN. The SU, acting as honeynode, refrains from transmitting its own data packets and instead transmits garbage data with specific transmission characteristics. Such transmission characteristics lure the attacker to jam the honeynode's transmission. For example, if an attacker targets the highest transmission power, then the honeynode transmits with highest possible power, while all other SUs keep their transmission power lower than the honeynode's power.

The description of the learning mechanism of honeynet is provided in [11]. When CR-honeynet is deployed, and the attacker is evolving, the attacks will sometimes be trapped by honeynode, and sometimes can strike legitimate SUs. We define one parameter, *attractiveness of honeynet* (ξ) as the probability that the honeynode is the one to be attacked, conditional on observing a jamming attack. Note that ξ depends on how well the CR-honeynet learning mechanism works. In this paper, our goal is to investigate the effectiveness of CR-honeynet with different values of ξ and determine when it is/not beneficial to deploy CR-honeynet.

2.4. Queue model

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

We can model an SU as a server with vacation where vacation is special service with higher priority. There are many mathematical models that deals with servers with vacations [22–25] where the server has the option to take vacations only at the end of its current service. Because the sensing period has deterministic length and intervals, the server model does not conform to the usual server with vacations. Instead, the sensing period acts as a "priority" customer whose inter-arrival rates and service times are deterministic. In this case a packet transmission has to be delayed if this packet can not be transmitted within the transmission period. In this paper we derive a mathematical model for server with deterministic vacations that portray the behavior of SUs in CR-Honeynet.

3. Mathematical Model of an SU

3.1. Queuing Characteristics

In earlier telecommunication networks, voice packets were generated at fixed rates or at fixed burst sizes [26, 27]. For this kind of system the inter-arrival time is fixed and the value depends on the codec (voice digitization technique) used

[26, 27]. Voice activity detection and Silence suppression techniques introduces randomness in packet arrival time.

With the increase in usage of multimedia applications on smart-phones the nature of the traffic flow is very complex to model. Because of the independence between sources, a memoryless inter-arrival time may be a good model. This observation is supported by statistical analysis. The studies carried out in various experiments [28–31] have concluded that when many different applications are merged, the packet arrival process tends to follow Poisson process. Fig. 4 provides a depiction of how packets from different applications flow to a queue. We use λ_i to denote the rate of the Poisson process of packet arrivals at SU labeled i, and $\{N_i(t); t \geq 0\}$ to denote the corresponding arrival process. When a single queue is analyzed, we drop the subindex i.

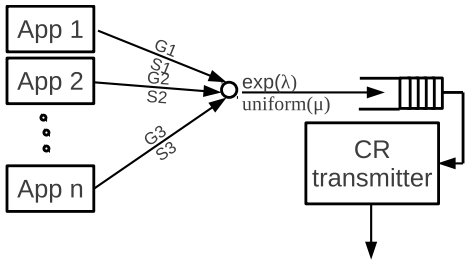


Figure 4: Depiction of how packets arrive to queue

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

During the transmission periods of length T_t, the model corresponds to a $M/G/1$ queue [23] where the service time of consecutive packets $\{S_n\}$ are independent and identically distributed. During the sensing periods of length T_s and transmission periods when the SU is chosen as a honeynode our server stops servicing the queue, which nonetheless continues to accumulate arriving packets. The effect of an attack during a transmission period when the SU is not a honeynode is that all packets transmitted in that slot are lost.

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

3.2. Queuing Model with Vacations

For simplicity, we assume that there are more free channels than the number of SUs in the CRN. In this section we assume that an SU is chosen to be "sacrificed" as a honeynode at every transmission period. If an SU is chosen as a honeynode, then all the new arriving packets join the queue and wait until the next transmission period, where the SU is not chosen as a honeynode. The analysis of this section assumes a random policy, where the i-th SU is chosen as honeynode with probability p_i, independently of past assignments and of

the state of the CRN. Other benchmark policies (such as round robin) will be discussed in later sections and compared via simulation experiments.

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

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

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

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

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

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

For a renewal process with no preemption, the distribution of this variable and its expectation can be calculated asymptotically [23]. In our model, where "many" services can be completed during transmission time (specifically, when $\ell_2 << T_t$) we can argue that $\tilde{S}$ will have this known asymptotic distribution as an approximate distribution, so that

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

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

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

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

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

calculate:

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

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

Using this approximation,

$$\mathbb{E}(X) = \frac{\rho\,\mathbb{E}(S^2)}{2\mathbb{E}(S)} \tag{1}$$

For any constant $a > 0$,

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

which yields:

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

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

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

which yields an implicit equation for μ':

$$\mu_i'(T_t + T_s) = \mu\left(\mu T_t - \frac{\mathbb{E}(S^2)\lambda_i}{2\mathbb{E}(S)\,\mu_i'}\right)(1-p_i) \tag{3}$$

Solving the quadratic equation (3) gives values of μ_i' that depend on λ_i.

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

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

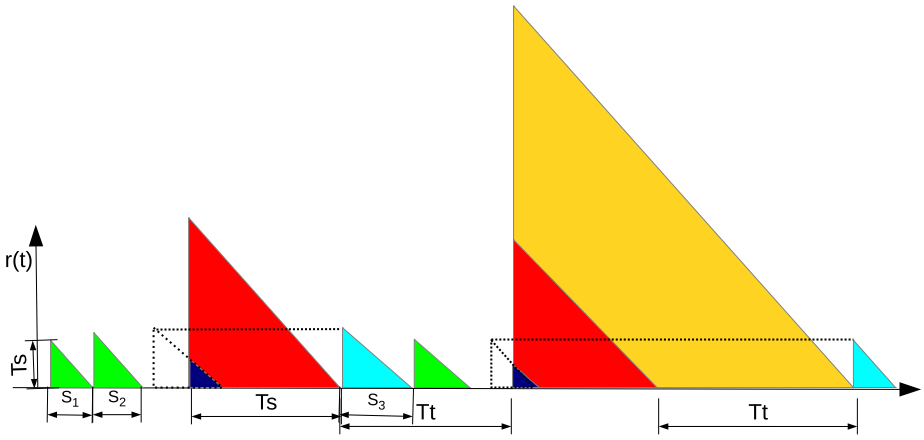


Figure 5: Path of the residual service of customer currently in service, for an SU. Here, green indicates packet transmission; red is Channel sensing; yellow is Serving as honeynode and blue indicates, the SU postpone current packet transmission as it can not be done within transmission period, cyan indicates postponed packet transmission.

Theorem 1. *Suppose that i'th SU has incoming rate λ, and that it is chosen as a honeynode independently of the state of the queue, with long term frequency of p. Furthermore, assume that this queue is stable and ergodic and let X satisfy equations (1) and (2). Then the stationary delay in queue is:*

$$W_q = \frac{R}{1 - \lambda \mathbb{E}(S)(1+\Delta)}, \tag{4}$$

where the stationary residual service time is:

$$R = \frac{\lambda\, \mathbb{E}(S^2)}{2} + \frac{\mathbb{E}(T_s + X)^2\,(1-p) + \mathbb{E}(T_s + T_t + X)^2\, p}{2(T_s + T_t)} \tag{5}$$

and the correction factor for the vacations is:

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

Proof. We use the residual service approach [22, 23] to calculate the stationary average delay in queue (assuming that it is well defined) as follows. Sensing periods of length T_s and honeynode periods of length T_t correspond to a "vacation" of the server, and are followed by transmission times of length T_t, during which consecutive packets with varying sizes enter service. Unlike the usual analysis of servers with vacations, here a vacation starts at deterministic times, and not necessarily at the end of busy periods. When not idle, the server can be in three different states: (a) a packet is being transmitted, (b) the server is on vacation, or (c) the current transmission is postponed and the server is waiting for the vacation.

Fig. 5 shows a typical path of the *residual* time until the completion of the current task (a service, a vacation, or the wait for the vacation), that we call

$r(t)$. Under ergodicity, the stationary average residual service is the same as the long term average, given by:

$$R = \lim_{t\to\infty} \frac{1}{t} \int_0^t r(t)\, dt = \lim_{t\to\infty} \frac{1}{t} \left[\sum_{i=1}^{M(t)} \frac{S_i^2}{2} + \sum_{i=1}^{V(t)} \frac{L_i^2}{2} \right] \tag{6}$$

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

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

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

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

where X satisfies (1) and (2). Applying these results in (6) gives expression (5).

The rest of the argument is as follows. It is a known property of Poisson processes that sampling a system at Poisson arrival epochs yields states that have a stationary distribution. This is sometimes called " a random snapshot" of the system. In queuing theory this property is also known as "PASTA" (Poisson arrivals see time averages). Using this property an arriving customer will encounter N_q customers in queue, where N_q is a random variable that has the stationary distribution of the queue length. The average wait time is thus the sum of the expected service time of the N_q customers in queue, plus R, plus the contribution of the vacation periods during the waiting time. Call T the required service time for the customers in queue, then using Little's Law:

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

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

$$W_q = \lambda W_q \mathbb{E}(S) + R + v_s(\, T_s + \mathbb{E}(X)) + v_h\, T_t, \tag{7}$$

where v_s and v_h are the (expected) number of sensing and honeynode periods (respectively) that fall within the time required to transmit all the N_q customers

in front of the new arrival. In the expression above we have used the fact that for every sensing period, the actual vacation time is not just T_s but we must add the lost time from the postponed service (if any). On average, the stationary contribution of this excess is $\mathbb{E}(X)$.

We now proceed to the calculation of v_s and v_h. In order to do so, we will use Wald's theorem [23]. Given T, the actual number of (true) transmission periods required to provide the service for the N_q customers in queue is:

$$\nu_t = \min\left(n:\ \sum_{i=1}^{n}(T_t - X_i) \geq T\right), \tag{8}$$

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

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

Rewrite (8) as: $\sum_{i=1}^{\nu_t} Z_i \leq T < \sum_{i=1}^{\nu_t+1} Z_i$ and take expectations to get:

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

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

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

Replacing now these values in (7) and using $\mathbb{E}(T) = \lambda W_q \mathbb{E}(S)$, we obtain

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

which yields (4), after some simple algebra.

4. Honeynode Selection Policies

4.1. State Dependent Policies for Uniform traffic distribution

When an SU is chosen as honeynode with initial queue size of Q packets, the queue size at the beginning of the following transmission period is $Q + A$,

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

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

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

This is called the "storage process" and it is dual to the surplus process in the canonical model for risk theory [32]. We are interested in evaluating the probability that the queue empties within the current transmission period, that is, $\mathbb{P}(\tau_u \leq T_t)$. This quantity is known in classical risk theory as the finite horizon "ruin probability". Because there are no closed form solutions, a number of methods have been proposed in the literature to evaluate the ruin probability, mostly when $T_t = \infty$.

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

$$\phi_N(j) = \mathbb{P}(\tau_{j\delta} \leq N\delta). \tag{9}$$

which defines the probability that the queue will empties within the next transmission period. Then we are interested in solving (9) for $N = \lfloor T_t/\delta \rfloor, j = \lfloor u/\delta \rfloor$. First notice that if $j \geq N$ then $\phi_N(j) = 0$, because it takes longer to serve the current packets than the prescribed time horizon. Next, suppose that $j < N$. Conditioning on the event that the first packet arrives during the interval $[(k-1)\delta, k\delta)$, it is immediate that $\phi_N(j) = 1$ for all $k > j$ (no arrivals while there is transmission, so the queue empties) which happens w.p. $e^{-\lambda j\delta}$. For $k \geq j$ the new arrival has Y bytes, and k bytes have been transmitted. The function ϕ satisfies the recursive equations:

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

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

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

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

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

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

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

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

$$\text{PDR}_i = \frac{1}{n}\left(1 - \frac{\xi}{n}\right). \tag{10}$$

This is verified in section 5.3.

Therefore, strategies for honeynode selection may include choosing the SU that has the largest probability of emptying its queue. For a CRN where all SU's have identical traffic (same arrival rates), choosing the SU with highest probability of emptying the queue is equivalent to choosing the SU with lowest queue size. Using, *largest probability of emptying queue* strategy, the CRN with uniform traffic chooses an SU that has the lowest queue at the beginning of a transmission period. For comprehensiveness, we compare this policy with round robin and random honeynode selection. In *random honeynode selection* strategy, one SU is chosen randomly to serve as a honeynode. In *round-robin honeynode selection* strategy, each SU takes a turn to serve as a honeynode in a cyclic order. In section 5.3, we present the system performance of these honeynode selection strategies and also compare the performance of CRN when it does not use a honeynet.

4.2. Optimal honeynode selection strategy for non uniform traffic distribution

So far we have considered uniform traffic load among all SUs in a CRN. In this case, state dependent policy of choosing SU with lowest queue size is beneficial in terms of overall system performance as can be seen in section 5.3. However, choosing SU with lowest queue size provides lowest fairness in the case of non-uniform traffic (i.e. SUs with different data rate requirements). It may easily be possible that the SU with lowest traffic is starved of services. This particular SU would be chosen as honeynode most of the time due to lower accumulated packets in the queue compared to other SUs. Repeatedly dedicating one SU over the other SUs results in more queuing delay for this SU. Round robin strategy provides fairness of service but lacks in overall system performance. Section 5.6 presents this trade-off.

Queuing delay and PDR define Quality of Service (QoS) measure of different applications. Some applications (such as real time application) can tolerate

packet loss but not delay and some applications (such as FTP) can tolerate delay but can not tolerate packet loss. A utility function has to be defined considering the delay and PDR so that CR-Honeynet can determine the best candidate for Honeynode. Utility function varies with the application. In section 5.5, we have used *R-score* as utility function for voice application. We have already stated that packet arrival model is a marked Poisson process where the marks specify the packet size. We can calculate transmission or service time (S_i) for each packet in the queue at the beginning of transmission period. Let's say U_i is the utility function of SU_i that depends on queuing delay and the PDR. U_i^{avg} is the average of U_i observed till the last transmission period. Now we model the honeynode selection strategy to a maximization problem.

$$
\begin{aligned}
\textit{maximize}: \quad & \sum_{i\in\mathbb{N}} \left(U_i(d_i^{exp}, \text{PDR}_i)\right)^2 \sum_{i\in\mathbb{N}} \psi(i).U_i^{avg} \qquad (11)\\
\textit{subject to}: \quad & \exists!\ i \in \mathbb{N}\ (\psi(i) = 1)\\
& d_i^{exp} = \frac{\frac{\left(\sum_{j=1}^{Q(i)} S_j + T_s + T_t.\psi(i)\right)^2}{2T_s+Tt}}{1-\lambda_i \mathbb{E}_i(S)(1+\Delta_i)},\\
& \Delta_i = \frac{T_s + \lambda_i \mathbb{E}(X)}{(T_t - \mathbb{E}(X))}.
\end{aligned}
$$

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

Determining the SU to select is very easy to calculate from the above mentioned maximization problem. We consider all SUs as possible candidates for honeynode and plug $\psi(i) = 1$ separately. After calculating the utility we choose the SU that calculates highest according to (11).

5. Simulation and Results

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

5.1. Simulation parameters and model

We coded a *discrete event simulation* [33], written in Python. in order to analyze CR-honeynet's performance. All arrival rates (λ) are in millisecond domain and mean λ packets per millisecond. As a first step, we consider equal arrival rates $\lambda_i = \lambda$ amongst the SUs. Under this assumption, the randomized and the queue-dependent policies become a randomized policy with equal

Table 1: Simulation Parameters

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

probabilities, and a minimum queue-size policy, respectively. The data for our model is given in Table 1.

In all our simulations, we use the technique of antithetic random variables (ARN) for increased precision. For the infinite queue model, where waiting and loss are monotone functions of the inter-arrival and service variables, ARN ensures variance reduction [33] (we used the inverse function method for generating random variables). For the finite buffer model, because some packets may be lost, it is no longer true that larger inter-arrivals (service times) always have a decreasing (increasing) effect on the delay. Although the theory does not ensure variance reduction for the finite buffer model, we verified this by experimentation.

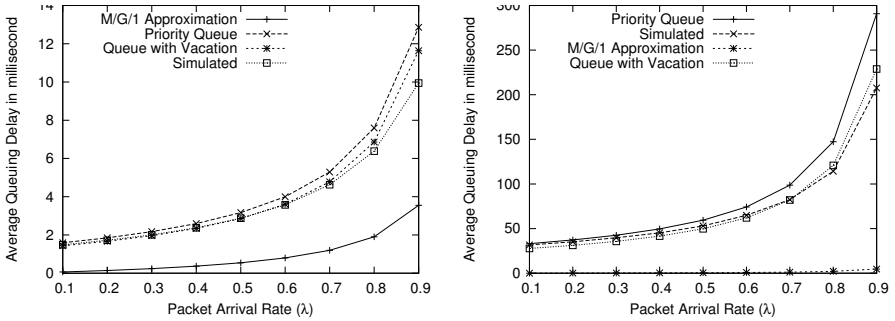
Using a simulated time of $50,000$ time slots entails that the number of packets served in each SU is also a random variable. For each replication of the simulation, we discarded the "warm up" data corresponding to the first 100 time slots. Preliminary simulations were used to choose these numbers, testing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

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

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

5.2. Comparison of approximations

Using parameters in Table 1, μ' can be calculated as in (3). When $\lambda \in [0.1, 0.9]$ and no honeynodes are assigned ($p_i = 0$) we get the range of values $\mu' \in [1.05513, 1.05551]$. For random honeynode assignment ($p_i = 1/N$), the corresponding range is $\mu' \in [1.00283, 1.00320]$, ensuring stability for all queues.



(a) without Honeynode (b) With one Honeynode

Figure 6: Average Queuing Delay for simple cognitive radio network

The analytical formulas available hold for the infinite buffer model and are as follows.

M/G/1 Queue. To obtain an expression for the stationary average delay or waiting time in the queue, we calculate a first crude approximation using the $M/G/1$ formulas with the effective rates λ and μ' [23].

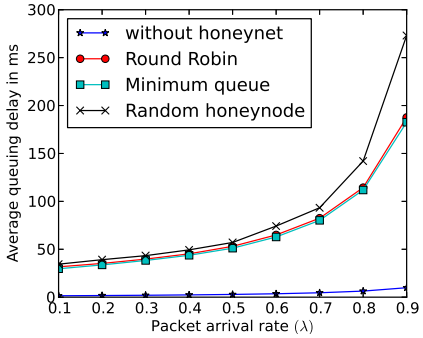
Priority model. A second approximation is based on a $M/G/1$ priority queue [23]. The sensing operation is to be served with higher priority, whereas packet transmission is a low priority job. Formula (12) estimates the stationary average queuing delay for a packet with service priority i, when all customer classes arrive according to independent Poisson processes.

$$W_q^i = \frac{\lambda_1 \mathbb{E}[S_1^2] + \cdots + \lambda_n \mathbb{E}[S_n^2]}{2 \prod_{j=i-1}^{i} (1 - \lambda_1 \mathbb{E}[S1] - \cdots - \lambda_j \mathbb{E}[Sj])} \tag{12}$$

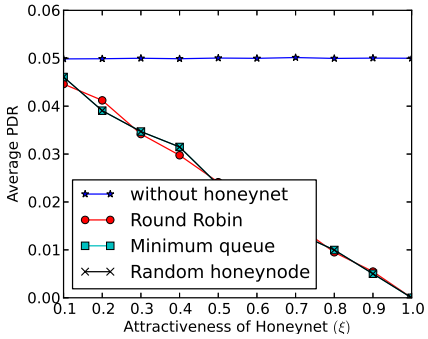
This formula is only an approximation because the arrival rate of the "sensing" or high priority jobs is $\lambda_1 = (T_s + T_t)^{-1}$, and $S_1 = T_s$ is deterministic. Second high priority job is serving as honeynode where $\lambda_2 = p(T_s + T_t)^{-1}$ and $S_2 = T_t$, while $S_3 \sim U(0.1, 1.7)$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

Vacation model. This corresponds to our formula (4). When no honeynodes are assigned, we use $p_i = 0$. For the random honeynode assignment we use $p_i = 1/20$.

We have simulated two scenarios. In the first scenario we assume there is no attacker, hence the CRN does not use honeynet. In the second scenario there is an attacker and the CRN dedicates an SU as honeynode in each transmission period. Fig. 6a and Fig. 6b show the results. We can clearly see that average queuing delay is higher for the second case as each SU serves as honeynode at its own slot and delay the packets. This degradation of performance is balanced by achieving lower packet drop while using CR-Honeynet to defend against jamming attack. We have presented these results as benchmark, only to compare the accuracy of our simulation model to other established queue model. The discrepancies are not very visible for smaller values of λ but they become more apparent for heavier traffic regimes, where our vacation formula seems to agree best with the simulated system.



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



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

Figure 7: Results for CRN with infinite buffer

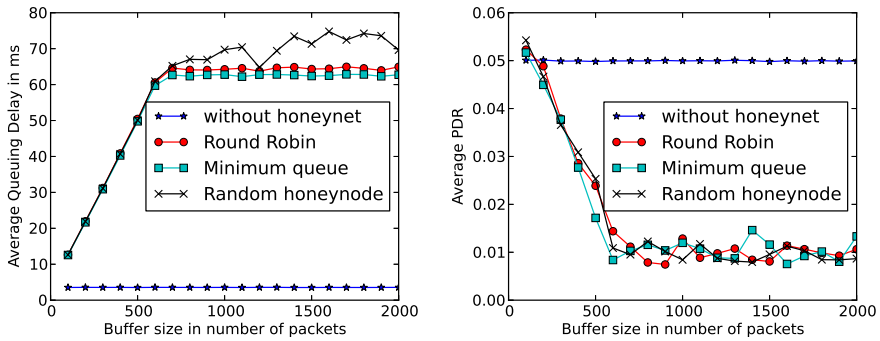
5.3. Comparison of honeynode assignment strategies for CRN with infinite buffer

In this section, we have used FTP data transfer where all SUs with uniform load. Fig. 7a shows the average wait per packet as λ increases with fixed $\xi = 0.8$ and infinite buffer sizes. In infinite buffer systems there is no packet drop for queue overflow, and therefore PDR is independent of λ. The only cause of packet drop is the jamming attack. Simulation results reflect that with $\xi = 0.8$, having no honeynode gives PDR of 0.05 and with one honeynode, PDR is 0.01 for all values of λ.

For the infinite buffer model, if θ_i is the probability that i'th SU is attacked and p_i is the long term fraction of periods where SU_i is chosen as a honeynode (assuming stationarity), then $\text{PDR}_i = \theta_i((1 - p_i) + p_i(1 - \xi_i))$. When $\theta_i = p_i = 1/N$ and $N = 20$ we obtain the linear function $0.05(1 - 0.05\xi)$ as verified in Fig. 7b. The attractiveness (ξ) does not affect the queue size, which is only dependent on the strategy and the incoming rate λ. Simulation result shows average queuing delay for no honeynode, random , minimum queue and round-robin selection strategies are 3.54 ms, 72.55 ms, 62.1ms and 64.62 ms respectively for all values of ξ. These results clearly say that state dependent policy i.e. selecting honeynode based on minimum queue length is performing better compared with others strategy. Honeynet ensures less packet drop at the cost of increased queuing delay.

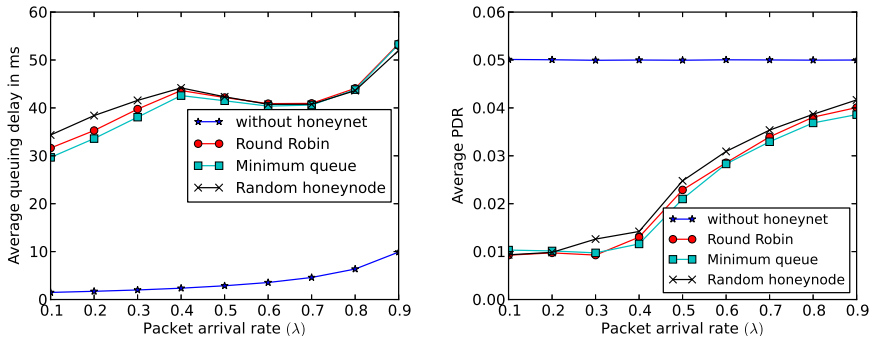
5.4. Performance of CRN with finite buffer and uniform traffic

When the queues have limited buffer capacity, incoming packets that can't fit in the buffer are dropped (lost). Even in the absence of attacks $\text{PDR}_i \neq 0$. In the absence of analytical models, we use simulations to assess the performance of various honeynode assignment strategies. Fig. 8a and Fig. 8b show the results for the average wait and PDR respectively, as a function of the buffer size, when $\lambda = 0.6$ and $\xi = 0.8$ are fixed. On average $\lambda \times (2T_s + T_t) = 630$ packets would be queued when SU serves as honeynode. There would be queue overflow if buffer is smaller, then SUs have significant PDR due to overflow. For larger buffer, performance is similar to that of an infinite buffer.



(a) Average Queuing Delay in ms (b) Average PDR

Figure 8: Results varying buffer size of SU with $\lambda = 0.6, \xi = 0.8$



(a) Average Queuing Delay in millisecond (b) Average PDR

Figure 9: Results for CRN varying λ with $\xi = 0.8$ and Buffer of 400 packets

We run another set of simulations with buffer size as 400 packets for every SU. Fig. 9a and Fig. 9b show the observed average queuing delay and PDR respectively. With low arrival rate λ (below a threshold) honeynode selection strategy based on minimum queue is performing better. This threshold value of λ should be Buffer Size$/(2T_S + T_t) = 0.381$ because one SU can accumulate this amount of packets when serving as honeynode. When λ goes above the threshold, and when the SU is serving as honeynode, it accumulates many packets so that the queue overflows which causes increase in PDR. Below this threshold SUs behave similar to infinite buffer model. These two graphs show a good trade-off between Delay and PDR. With limited buffer and from the two figures it is clear that the system administrator have to come to a conclusion at particular value of λ and ξ and specified buffer size, whether to apply a honeynet or not. At higher value of λ and loss tolerant traffic (such as real time video) not having honeynode as it can not tolerate delay.

5.5. When Honeynet can be applied and when not

We have seen that the choice of dedicating SUs as honeynode comes with a trade-off. We need to analyze when CR-Honeynet is beneficial to use or

not. During each transmission period, the CRN assigns an SU as a honeynode. This decreases overall PDR while introducing extra delay to packet transmission. From (10), we can see that, for the case of uniform traffic (i.e. all SUs handle equal traffic load), PDR is inversely proportional to the attractiveness of honeynet (ξ). Again, an increase in PDR causes a degradation in the QoS of the system. In this section, we want to determine the threshold, the lowest value of attractiveness for effective honeynet (ξ^*). ξ^* is the point at which the net gain of using honeynet is zero. A CRN achieves a higher performance by assigning honeynode when $\xi > \xi^*$. A honeynet, with effective attractiveness below ξ^*, is not worth of dedicating one SU to serve as honeynode. To determine this threshold, we need to define a performance function of the CRN that takes into account both delay and PDR. *Non real time traffic*, such as FTP can tolerate delay provided all packets are received successfully. A honeynode with $\xi > 0$ is always beneficial for non real-time traffic because it guarantees lower PDR, compared to not having a honeynode. So, we try to find out ξ^* for *real time traffic* that has a stringent end-to-end packet delay requirement. Higher end-to-end delays degrade the system performance significantly and a packet is considered lost if the end-to-end delay exceeds a certain threshold.

As the first step in analyzing real time traffic, we consider Voice over IP (VoIP) traffic to determine ξ^*. For VoIP services, there are two indicators that define QoS, namely *mean opinion score* (MOS) and *R-score* [34, 35]. Both of these indicators depend on end-to-end delay and packet loss. The end-to-end delay or mouth-to-ear delay of voice application is, in turn, composed of three parts, the codec delay (d_{codec}), the playout delay ($d_{playout}$), and the network delay ($d_{network}$). Codec delay and playout delay are dependent on the codec being used and the receiver side buffer respectively. These delays are usually very small (generally between 10 and 50 ms). The network delay is the component that varies with network conditions. Again, the network delay consists of queuing delay (d_{queue}) and transmission delay ($d_{transmission}$). Transmission delay, or the time taken to transmit on packet, is very negligible (0.1 to 1.7 ms). The sensing period, as well as dedicating an SU as a honeynode, introduces a very high queuing delay to the packet transmission. Considering all these factors, end-to-end delay can be written as:

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

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

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

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

$$\text{MOS} = 1 + 0.035R + 7 \times 10^{-6} R(R - 60)(100 - R) \quad (13)$$

$$R = 94.2 - (\gamma_1 + \gamma_2 \ln(1 + \gamma_3 e)) - (0.024d + 0.11(d - 177.3)\mathbf{H}(d - 177.3) \quad (14)$$

Table 2: Coefficient parameters for calculating loss impairment [34–38]

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

Where, $\mathbf{H}(x)$ is an indicator function. $\mathbf{H}(x) = 0$ if $x < 0$ and 1 otherwise. γ_1, γ_2 and γ_3 are Loss Impairment Parameters that depend on specific codecs that are used for digitization and packetization of voice samples. Coefficient parameters for useful codecs are provided in Table 2, which provide the application layer data rate. Every packet that passes through MAC layer has to contain the transmission layer, network layer, and MAC layer headers.

Cole *et al.* [35] have provided a table that signifies the QoS with MOS values which indicates, the higher MOS value, the better the QoS. Again, when MOS is plotted against R-score, it reveals that a system obtains better QoS when R-score is higher. R-score of 80 and above is desired whereas 70 and higher is acceptable. We are taking R-score as the CRN's performance measure and the goal is to maximize the average R-score.

Now, we build another simulation of CRN where all SUs, with infinite buffer, are transferring packets in accordance with G.711 codec [34], for voice digitization. Here, we are considering the minimum queue honeynode selection strategy. Fig. 10 plots the average R score for the CRN. From (4), we see that the queuing delay is irrespective of ξ, where as (10) reveals that PDR is inversely proportional to ξ. When all other parameters are unchanged, R-score is proportional to ξ. In other words, an increase in ξ enhances the QoS, which can be seen in the figure. We say, ξ^* is the attractiveness of honeynet for which, $R_{with\ honeynode} = R_{without\ honeynode}$. When $\xi > \xi^*$, the CRN dedicates one SU as a honeynode. When using no honeynode, we observe a R-score of 73.11. We can clearly see here that, for $\xi = 0.44$, the R-score of using honeynet is same as of not using honeynet. For this particular CRN, we can conclude that the lowest effective attractiveness of honeynet $(\xi^*) = 0.44$.

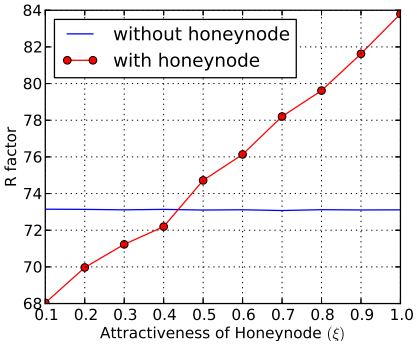
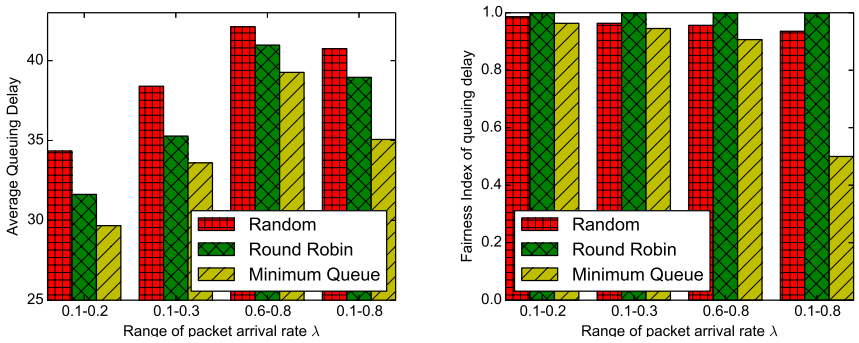


Figure 10: Average R-score of the CRN

5.6. Fairness of performance for non uniform real time traffic

We consider Jain Fairness index [39] to measure how fair our honeynode selection strategies are. For a network of n nodes, if observed values of a performance parameter are $x_1, x_2, \ldots x_n$ for n nodes, then fairness index is defined as



(a) Average queuing delay (b) Fairness Index among all SUs

Figure 11: Simulation results for CRN with non uniform traffic.

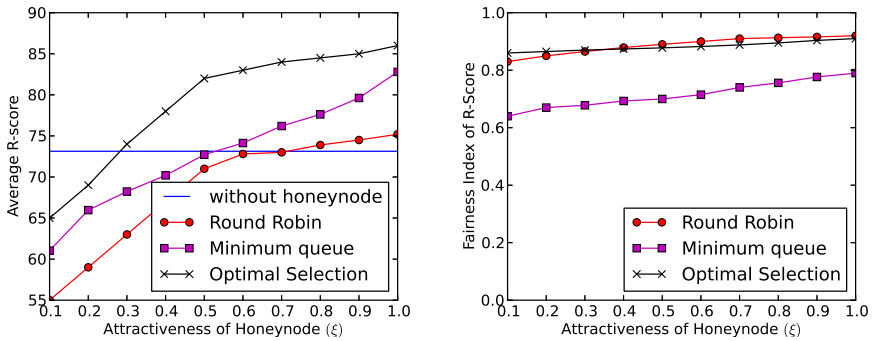
$\frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n\sum_{i=1}^{n} x_i^2}$. Fig. 11a and Fig. 11b depicts the overall average queuing delay for the CRN and fairness index of queuing delay, respectively. All the SUs choose a λ randomly from the range given in the X-axis. For example, in the third simulation, all SUs have λ in between 0.6 and 0.8. The first three simulation sets have lower variances of λ among SUs. The last set of simulations have higher variances of λ. Here we consider FTP data transfer as the application. We can clearly see that the minimum queue honeynode selection strategy provides a lower queuing delay, compared to other honeynode selection strategies. However, the minimum queue honeynode selection strategy performs very poorly, in terms of fairness. Actually, some SUs get better transmission by making the SUs that have lower λ to starve of packet transmission.

5.7. Performance of CRN for real time non-uniform traffic

In this section we study how the CRN can achieve fairness for all SU's utility. We have already stated that utility or system performance for real-time traffic is dependent only on PDR as it is not stringent to delay. Eq.10 shows that PDR only depends on ξ. Now, ξ depends on the efficiency of the learning mechanism used in the honeynet. Selection of honeynode does not have any effect on ξ or PDR. So, for real-time traffic selecting honeynode with minimum queue strategy is optimal as it achieves the lowest average queuing delay and thus highest utility.

As a first step towards analyzing CRN with non-uniform traffic, We consider VoIP traffic to study the performance of real-time-traffic. We run sets of simulation where all SUs in the CRN transmit VoIP data in accordance with different voice codecs chosen randomly from the Table 3. For simplicity in the simulation we have excluded the codecs with active voice detection and silence suppression. All codecs demand different throughput and have different packet arrival rates.

We have proposed a optimal honeynode selection algorithm in section 4.2. For VoIP traffic we consider R-Score as the performance measure or utility function in (11). It is interesting to know that all codecs result in different R-score for same PDR and delays. Fig. 12a shows the average R-score observed



(a) Average R-Factor among all SUs (b) Fairness Index of all SUs

Figure 12: Comparison of performance for all honeynode selection strategies

Table 3: VoIP packet characteristics [40]

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

for the CRN which reveals that our proposed optimal strategy is performing better compared to other honeynode selection strategies. Here we can see that CRN without honeynode results in R-Score of 73.13. We observe that the lowest effective attractiveness (ξ) are 0.283, 0.53, 0.71 for optimal selection, minimum queue selection and round robin selection respectively. Fig. 12b plots the fairness index for honeynode selection strategies. We can see that minimum queue selection strategy performs very poorly while the optimal selection and round robin are pretty fair. Round robin obtains lower fairness index in this simulation as different codecs provide different R-Score even if the SUs achieve same queuing delay. So, we can conclude that the optimal honeynode selection strategy is performing well for real-time VoIP traffic.

6. Conclusions and future work

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

time traffic our proposed honeynode selection strategy provides optimal system performance while maintaining fairness.

7. Appendix

Table 4: Useful notations used in the paper

Notations	Meaning
T_s	sensing period
T_t	transmission period
λ	rate of the Poisson process of packet arrivals
Y	uniform random variable of packet size
S	uniform random variable of packet transmission time
l_1	lower bound of packet size
l_2	higher bound of packet size
X	fraction of service postponed due to sensing interval
ρ	probability that SU is busy for sensing or honeynode duty
μ	packet transmission rate during transmission period
μ''	packet transmission rate considering vacation intervals
W_q	stationary average queuing delay
Δ	correction factor for vacation
R	stationary residual service time
PDR	packet drop rate
θ_i	long term probability that SU_i is attacked
p_i	long term probability that SU_i is chosen as honeynode
v_s	expected number of sensing period before queue empties
v_h	expected number of honeynode duties before queue empties
N	total number of SUs in the CRN
d_i	delay experienced by SU_i

References

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

[2] IEEE draft standard for information technology -telecommunications and information exchange between systems - wireless regional area networks (WRAN) - specific requirements - part 22: Cognitive wireless ran MAC & PHY specifications: Policies and procedures for operation in the TV bands, IEEE P802.22/D2.0 (2011).

[3] S. Bhattacharjee, S. Sengupta, M. Chatterjee, Vulnerabilities in cognitive radio networks: A survey, Computer Communications 36 (13).

[4] T. X. Brown, J. E. James, A. Sethi, Jamming and sensing of encrypted wireless ad hoc networks, in: Proceedings of the ACM international symposium on Mobile ad hoc networking and computing, 2006, pp. 120–130.

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

[6] V. Navda, A. Bohra, S. Ganguly, D. Rubenstein, Using channel hopping to increase 802.11 resilience to jamming attacks, in: 26th IEEE International Conference on Computer Communications (INFOCOM), IEEE, 2007.

[7] W. Xu, K. Ma, W. Trappe, Y. Zhang, Jamming sensor networks: attack and defense strategies, Network, IEEE (2006) 41–47.

[8] C. Sorrells, P. Potier, L. Qian, X. Li, Anomalous spectrum usage attack detection in cognitive radio wireless networks, in: IEEE International Conference on Technologies for Homeland Security (HST), 2011, IEEE, 2011, pp. 384–389.

[9] C. Popper, M. Strasser, S. Capkun, Anti-jamming broadcast communication using uncoordinated spread spectrum techniques, Selected Areas in Communications, IEEE Journal on 28 (5) (2010) 703–715.

[10] S. Misra, S. K. Dhurandher, A. Rayankula, D. Agrawal, Using honeynodes for defense against jamming attacks in wireless infrastructure-based networks, Computers and Electrical Engineering 36 (2) (2010) 367 – 382.

[11] S. Bhunia, S. Sengupta, F. Vazquez-Abad, Cr-honeynet: A learning & decoy based sustenance mechanism against jamming attack in crn, in: Military Communications Conference (MILCOM), 2014 IEEE, IEEE, 2014, pp. 1173–1180.

[12] J. Burbank, Security in cognitive radio networks: The required evolution in approaches to wireless network security, in: 3^{rd} International Conference on Cognitive Radio Oriented Wireless Networks and Communications., 2008, pp. 1 –7. doi:10.1109/CROWNCOM.2008.4562536.

[13] Wi-spy spectrum analyzer, http://www.metageek.net/products/wi-spy/.

[14] GNU Radio, http://gnuradio.org/redmine/projects/gnuradio/wiki.

[15] Usrp kit, https://www.ettus.com/product/details/UN200-KIT.

[16] K. Pelechrinis, M. Iliofotou, S. V. Krishnamurthy, Denial of service attacks in wireless networks: The case of jammers, Communications Surveys & Tutorials, IEEE 13 (2) (2011) 245–257.

[17] V. Chatzigiannakis, G. Androulidakis, K. Pelechrinis, S. Papavassiliou, V. Maglaris, Data fusion algorithms for network anomaly detection: classification and evaluation, in: Networking and Services, 2007. ICNS. Third International Conference on, IEEE, 2007, pp. 50–50.

[18] C. Sorrells, L. Qian, H. Li, Quickest detection of denial-of-service attacks in cognitive wireless networks, in: Homeland Security (HST), 2012 IEEE Conference on Technologies for, IEEE, 2012, pp. 580–584.

[19] T. X. Brown, A. Sethi, Potential cognitive radio denial-of-service vulnerabilities and protection countermeasures: A multi-dimensional analysis and assessment, Mobile Networks and Applications 13 (5) (2008) 516–532.

[20] S. Anand, S. Sengupta, K. Hong, K. Subbalakshmi, R. Chandramouli, H. Cam, Exploiting channel fragmentation and aggregation/ bonding to create security vulnerabilities, IEEE Transactions on Vehicular Technology.

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

[22] K. C. Madan, A. Z. Abu Al-Rub, On a single server queue with optional phase type server vacations based on exhaustive deterministic service and a single vacation policy, Applied Mathematics and Computation 149 (3) (2004) 723–734.

[23] S. M. Ross, Introduction to Probability Models, 10^{th}Ed., Academic Press.

[24] G. Choudhury, L. Tadj, An M/G/1 queue with two phases of service subject to the server breakdown and delayed repair, Applied Mathematical Modelling 33 (6) (2009) 2699–2709.

[25] S. Wang, J. Zhang, L. Tong, Delay analysis for cognitive radio networks with random access: a fluid queue view, in: INFOCOM, 2010 Proceedings IEEE, IEEE, 2010, pp. 1–9.

[26] L. X. Cai, X. Shen, J. W. Mark, L. Cai, Y. Xiao, Voice capacity analysis of wlan with unbalanced traffic, Vehicular Technology, IEEE Transactions on 55 (3) (2006) 752–761.

[27] L. Sun, E. Ifeachor, Voice quality prediction models and their application in voip networks, IEEE Transactions on Multimedia 8 (4) (2006) 809 –820. doi:10.1109/TMM.2006.876279.

[28] V. Grout, S. Cunningham, D. Oram, R. Hebblewhite, A note on the distribution of packet arrivals in high-speed data networks., in: ICWI, Citeseer, 2004, pp. 889–892.

[29] R. Jain, S. A. Routhier, Packet trains-measurements and a new model for computer network traffic, IEEE Journal on Selected Areas in Communication 6 (4) (1986) 986–995.

[30] M. Wilson, A historical view of network traffic models, http://www.cse.wustl.edu /∼ jain/cse567-06/ftp/traffic_models2.

[31] D. R. Dennis Guster, R. Sundheim, Evaluating computer network packet inter-arrival distributions, Encyclopedia of Information Science & Technologydoi:10.4018/978-1-60566-026-4.ch232.

[32] R. Kaas, M. Goovaerts, J. Dhaene, M. Denuit, Modern actuarial risk theory, Vol. 328, Springer, 2001.

[33] S. M. Ross, Simulation, 5th Edision, Academic Press, 2012.

[34] S. Sengupta, M. Chatterjee, S. Ganguly, Improving quality of voip streams over wimax, Computers, IEEE Transactions on 57 (2) (2008) 145–156.

[35] R. G. Cole, J. H. Rosenbluth, Voice over IP performance monitoring, ACM SIGCOMM Computer Communication Review 31 (2) (2001) 9–24.

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

[37] J. Q. Walker, Assessing voip call quality using the e-model.
URL http://www.recursosvoip.com/docs/english/AssessingVoIPCallQualityUsingtheE-model.pd

[38] V. Balan, L. Eggert, An experimental evaluation of voice-over-ip quality over the datagram congestion control protocol, School of Engineering and Scienze International University Bremen.

[39] R. Jain, D.-M. Chiu, W. R. Hawe, A quantitative measure of fairness and discrimination for resource allocation in shared computer system, Eastern Research Laboratory, Digital Equipment Corporation, 1984.

[40] Voice Over IP - per call bandwidth consumption.
URL http://www.cisco.com/c/en/us/support/docs/voice/voice-quality/7934-bwidth-consume.h