Keeping up with the Jammers: Observe-and-Adapt Algorithms for Studying Mutually Adaptive Opponents[2]

Bruce DeBruhl[a,1], Patrick Tague[a,2]

[a] *Carnegie Mellon University - Silicon Valley, NASA Research Park, Bldg. 23 (MS 23-11), P.O. Box 1. Moffett Field, CA 94035-0001*

Abstract

Securing the wireless medium is essential to provide the ubiquitous wireless services that we desire. Many studies have explored adaptive attackers and defenders but few have explored the interaction when both players adapt. In this work, we explore the design of an adaptive defender and attacker using an observe-and-adapt strategy. We simulate these algorithms and explore the interaction of adaptive players in two different jamming games. We show that when only one player adapts they improve their performance but when both players adapt the outcome is often reflective of biases in the game.

Keywords: Adaptation, Denial-of-Service, Communication Security, Jamming

1. Introduction

We depend on wireless communication systems in many aspects of our daily life. The wireless medium is completely open to anyone within a broadcast range, allowing for any system to broadcast on the wireless ether. The open nature of the wireless medium opens its users to a wide variety of attacks from malicious users. Examples of wireless attacks include spoofing legitimate users, denying service to legitimate nodes, or eavesdropping [1]. These attacks can focus on a whole network of nodes or on a single link in the network.

A lot of work has been done in the field of wireless denial-of-service (DoS) attacks [2, 3]. Examples of (DoS) attacks include interfering with the MAC protocol, interfering with the physical layer transmission, or interfering with higher layer protocols. Many interesting studies have looked at how DoS attacks can be made more damaging to a legitimate system. For example, optimal attack design [4], adaptive attack design [5, 6], and targeted attack designs [7] have all been studied. Optimal attack design [4] suggests balancing attack

Email addresses: debruhl@cmu.edu (Bruce DeBruhl), tague@cmu.edu (Patrick Tague)
[1]Telephone - 734-718-4023.
[2]Telephone - 650-335-2827.

impact with not being detected using game theory. Adaptive attack designs [5, 6] use information observed from the system to balance the impact and cost of an attack. Targeted attack design [7] looks to attack only selected nodes in a reactive manner. The understanding of advanced attacks is important to allow for design of secure and robust communication systems.

The ability of a defender to mitigate DOS attacks has also been explored. An early example of this is spread spectrum techniques at the PHY layer [1], which use shared secrets to adapt frequency usage and mitigate jamming. The shared secret concept was further explored in SPREAD [8] which uses a synchronized hopping technique over the whole communication stack to allow for increased resilience. Improvements to spread spectrum have also been explored at the local link by considering secure key distribution [9] and anti-jamming filtering [10, 11]. Researchers have also proposed protocol redesigns at the PHY and MAC layer with jamming attack resilience in mind, for example tuning rate adaptation, carrier sensing thresholds [12], frame masking, and packet fragmentation [13]. These works look to modify or optimize common protocols to mitigate the effects of jamming without modifying hardware. Optimal defense and attack strategies have also been studied using game theory [14, 15].

How an adaptive attacker and defender interact has many interesting implications. If we can provide bounds on adaptation needs and randomness to foil a set of attackers this increases our ability to guarantee communication resiliency. In this work, we introduce a framework that characterizes the effect of adaptation on secure communication systems. Our framework allows for players to be fixed, adaptive, transient, or random. We assume that when a player is adaptive their goal is to maximize the impact on the network's performance while minimizing their energy usage. We also assume that the adaptation is done by using observations. The adaptive player is not omniscient, knowing the system and their opponents algorithms, but rather has to observe-and-adapt on the fly. The minimal energy assumption is made because of the increased prevalence of battery powered devices like wireless sensor nodes and mobile phones. An example of why the attacker's energy constraint is interesting is the implications of using a mobile phone as an attacker. We make the following contributions towards the goal of understanding the interactions between attacking and defending players.

- We introduce a framework to analyze the interaction of an adaptive attacker and defender.
- We design algorithms that use an observe-and-adapt strategy to increase a player's performance.
- We design two games to test our algorithms and explore the interactions of an adaptive attacker and defender.
- We test the interaction of an adaptive attacker and defender and explore the outcome of the two-player adaptive game.

The remainder of this article is organized as follows. In Section 2, we introduce our model and systems. In Sections 3 and 4 we introduce our adaptation

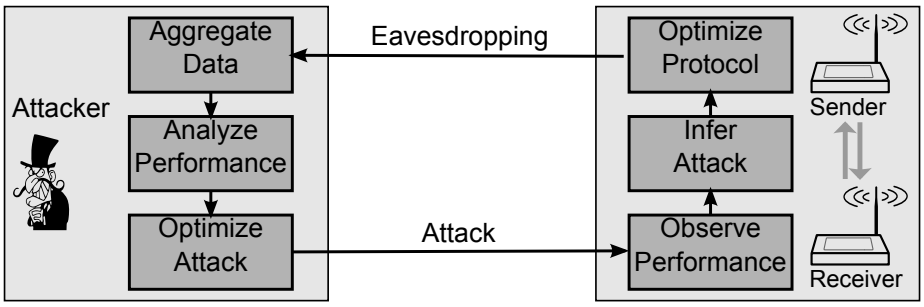


Figure 1: In this figure, we show our system including an attacker and defender which both use an observe and adapt parameters.

strategies for the one-player and two-player games respectively. We introduce two jamming games and show simulation results for our algorithms in Section 5. Lastly, we discuss the recent literature on adaptive and optimal security in Section 6 and conclude our paper in Section 7.

2. Model

In this work, we explore a system with a pair of legitimate nodes that communicate over a wireless medium and a malicious node that attempts to interrupt the communication between them. Since the legitimate nodes attempt to communicate while averting an attack, we collectively refer to them as the defender. Likewise, we refer to the malicious node as the attacker. Collectively, We refer to the attacker and defender as the players.

We consider the case where both the defender and attacker are energy-constrained. Due to the energy constraints, both players attempt to maximize their performance while minimizing their energy usage, effectively presenting each player with a multi-objective optimization problem. An energy-constrained player is easily motivated by many use cases including wireless sensor networks, ad hoc networking, smartphones, vehicular networking, and wireless infrastructure.

We propose an on-line *observe-and-adapt* model for both players. In this model, players use observations about their performance and their opponent's actions to infer the utility of their actions and to assist in making future decisions. The inferred performance information is then used to adapt the player's parameters, attempting to find better parameters. We show a high level overview of our system in Figure 1. This figure shows an attacker who eavesdrops to infer how well it performs and uses these observations to optimize its attack. Likewise, the defender observes its performance and the attack and uses these observations to optimize its protocol. As cognitive radio [16] and similar observation based communication protocols advanced this type of observe-and-adapt strategy becomes very feasible.

In the remainder of this section, we introduce assumptions about the players and their abilities Since digital communication systems are packet based in nature, we assume that the players both choose protocols and parameters in discrete time.

2.1. Players

Defender - The defender aims to maintain availability and dependability of communication with minimal energy usage. We assume that the defender only attempts to optimize one layer of the communication stack, though our work can be extended to coordinated multilayer optimization. To do this the defender defines the set of all protocols that they are equipped for at the layer to be optimized as $\mathcal{S}$. The use of a robust set of protocols for a defender is easily justified given the number of radios on a modern smartphone, showing that many devices have a plethora of protocols available. For the k^{th} time period a defender chooses a protocol $S_d^k \in \mathcal{S}$. Since most protocols have a set of tunable parameters, a defender also chooses a vector $\boldsymbol{\phi}_d^k$ as the parameters corresponding to a protocol S_d^k. For instance, if a defender is attempting to optimize its physical layer protocol, it could define its possible strategies as direct sequence spread spectrum and frequency hopping spread spectrum, $\mathcal{S} = \{\text{DSSS}, \text{FHSS}\}$. If at time k the defender select $S_d^k = \text{FHSS}$, then the parameter vector $\boldsymbol{\phi}_d^k$ comprises the number of channels N_C^k and hopping time t_h^k, yielding

$$\boldsymbol{\phi}_d^k = \begin{pmatrix} \text{number of channels} \\ \text{hopping time} \end{pmatrix} = \begin{pmatrix} N_C^k \\ t_h^k \end{pmatrix}. \tag{1}$$

Attacker - We also consider an attacker with many radios and define the set of possible attacks as $\mathcal{T}$ which includes the option not to attack $\mathcal{T}_0$. At any given time k an attacker chooses to mount an attack $S_a^k \in \mathcal{T}$. Since most attacks have tunable parameters, we define a parameter vector ϕ_a^k associated with S_a^k.

Energy Consideration - Since both players are energy-constrained, they are interested in the cost of a given protocol and parameter vector. We suppose that the energy used by the defender for a given time period k is a function of the protocol S_d and parameters $\boldsymbol{\phi}_d^k$. Thus we define the defender's energy consumption for a time period as $E_d^k = H_d(S_d^k, \boldsymbol{\phi}_d^k)$. Likewise, we define the attacker's energy consumption as $E_a^k = H_a(S_a^k, \boldsymbol{\phi}_a^k)$.

Interaction - We desire to understand the interaction of the attacker and defender. One way to do this is to define a dependability parameter and use this as a measure of the attacker's effect on the system and the defender's resilience to an attack. The dependability of the defending system is a function of both players' protocols and parameter vectors, so we define our dependability parameter over the time period k as $v^k = G(S_d^k, \boldsymbol{\phi}_d^k, S_a^k, \boldsymbol{\phi}_a^k)$. We assume that the attacker desires to minimize v and the defender aims to maximize v.

Since the players are interested in controlling system dependability with minimal energy usage, we define utility functions for the attacker and defender as u_a^k and u_d^k, respectively. We use a normalized weighted average method [17]

to combine the effect of the dependability parameter and energy usage for both players. We define the attacker's utility for time period k as

$$u_a^k = w_{E,a}\left(1 - \frac{E_a^k}{E_{a,max}}\right) + w_{v,a}\left(1 - \frac{v^k}{v_{max}}\right) \tag{2}$$

and the defender's utility function as

$$u_d^k = w_{E,d}\left(1 - \frac{E_d^k}{E_{d,max}}\right) + w_{v,d}\frac{v^k}{v_{max}}, \tag{3}$$

where $w_{E,e}, w_{v,a}, w_{E,d}, w_{v,d} \geq 0$ are weighting values satisfying $w_{E,a} + w_{v,a} = 1$ and $w_{E,d} + w_{v,d} = 1$.

The goal of the players thus becomes to optimize their respective utility functions. The utility functions are designed so that both players aim to maximize their utility. In the next section, we define inference and adaptation techniques that players use to optimize their utility functions.

2.2. Learning and Adaptation

The general process for learning and adapting is the same for either player so we discuss this process from one players perspective. The learning algorithms we consider in this work assume a finite number of options for both protocols and parameters. This is a natural assumption, since a node only has so many radios and digital parameters. We can assume, without loss of generality, that the protocol is uniquely determined by the parameters $\boldsymbol{\phi}^k$. This assumption allows us to design our approach considering the parameters alone. Therefore, we simplify our model to use dependability $v^k = G(\boldsymbol{\phi}_d^k, \boldsymbol{\phi}_a^k)$ and energy consumption $E_a^k = H_a(S_a^k, \boldsymbol{\phi}_a^k) = H_a(\boldsymbol{\phi}_a^k)$ and $E_d^k = H_d(S_d^k, \boldsymbol{\phi}_d^k) = H_d(\boldsymbol{\phi}_d^k)$.

2.2.1. Learning

Our model allows for players to make observations about their effect on the system as well as the previous plays their opponents have made. The players in our system observe their utility with respect to their parameters and their opponent's parameters. Each player then uses these observations to develop an estimate for the corresponding utility function. We denote the estimated utility function for the attacker as $\tilde{u}_a^k(\phi_a, \phi_d)$.

The history of the player's opponent's parameter choice is estimated as a weighting vector based on what the player's opponent has done in the past. The attacker's estimated weighting vector over the defender's parameter space $\boldsymbol{\Phi}_d$ is denoted as $\tilde{\mathbf{W}}_d(\boldsymbol{\Phi}_d)$.

For ease of illustration, we refer to the collective information about utility of parameters and the estimated probability of an opponent choosing a parameter as their knowledge. Thus, the more knowledge a player has the more likely they are to be able to optimize their parameters. However, if their opponent changes parameters, their previous knowledge can be detrimental and lead to sub-optimal parameter choices. We illustrate the learning process of a attacker

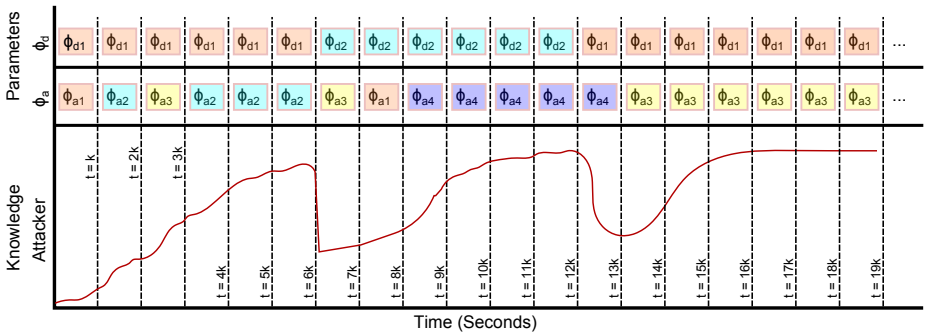


Figure 2: In this figure, we illustrate an adaptive attacker who makes observations and adapts its parameters. The attacker uses knowledge from previous observations to optimize its performance. However, when the defender changes parameters this knowledge can be detrimental and cause the attacker to make sub-optimal parameter options.

in Figure 2. The attacker finds an good protocol at the beginning, but when the defender changes protocols it must forget some history and relearn a knew strategy. We look at simulations of an adaptive attacker and defender interacting in Section 5.

2.2.2. Adaptation

As mentioned previously, this work considers adaptation of parameters and assumes that protocols are chosen a priori. Using the information gained from learning, the players choose parameters in such a way as to give a high probability of achieving a high utility. Each player may choose what are estimated as sub-optimal strategies at times to test how the system has changed.

One important aspect affecting how this observation and adaptation occurs is the relation of the adaptation rate of the two players. We define a *fixed* player as one who does not change parameters. We define a *transient* player as one who adapts at a rate much slower than its opponent; for example if the attacker is transient it adapts much slower than the defender. We define an *adaptive* player as one who adapts parameters at the same rate or faster than the opponent.

3. One Adaptive Player Formulation

In this section, we introduce two observe-and-adapt algorithms for an adaptive player interacting with a fixed or transient player. For ease of discussion and notation, we assume that the attacker is adaptive and the defender is fixed. This derivation is easily reversed for an adaptive defender and fixed attacker. Thus, we present how an attacker makes observations about its utility and uses these observations to adapt its attack parameters to minimize its utility function. We present two algorithms for how this could be done which we call *weighted observation* and *universal approximation*. The weighted observation algorithm uses observations of empirical performance to construct a weighting

$\tilde{\mathbf{u}}_a(\mathbf{\Phi}_a) \leftarrow \mathbf{1}$
for $k = 1 : \infty$ **do**
  $\mathbf{W}_a(\mathbf{\Phi}_a) \leftarrow \frac{e^{\kappa \tilde{\mathbf{u}}_a(\mathbf{\Phi}_a)}}{\|e^{\kappa \tilde{\mathbf{u}}_a(\mathbf{\Phi}_a)}\|}$
  $\boldsymbol{\phi}_a^k \leftarrow \text{rand\_sample}(\mathbf{\Phi}_a, \mathbf{W}_a(\mathbf{\Phi}_a))$
  $\tilde{\mathbf{u}}_{\mathbf{a}}(\boldsymbol{\phi}_a^k) \leftarrow \lambda \tilde{\mathbf{u}}_{\mathbf{a}}(\boldsymbol{\phi}_a^k) + \beta u_a^{obs}(k)$
end

algorithm 1: One-Player Weighted Observation Algorithm

vector which estimates the relative utility of different parameter choices. This vector is then used to weight a random sample of parameter values. The second algorithm uses universal neural network approximation [18] to estimate the utility function and optimize their strategy from a small set of observations. For both of these algorithms, we present techniques to also make them viable for the transient case.

3.1. Weighted Observation

We first introduce the weighted observation algorithm for one-player adaptation. This algorithm uses a weighted record of past observations and random sampling to select a parameter set that has, on average, a higher utility than a random parameter set. We first, define the estimated utility from a given parameter $\boldsymbol{\phi}_a^j$ as $\tilde{u}_a(\boldsymbol{\phi}_a^j)$. We then use this to construct a vector of all the estimated utilities as

$$\tilde{\mathbf{u}}_a(\mathbf{\Phi}_a) = \begin{pmatrix} \tilde{u}_a(\boldsymbol{\phi}_a^1) \\ \tilde{u}_a(\boldsymbol{\phi}_a^2) \\ \vdots \\ \tilde{u}_a(\boldsymbol{\phi}_a^m) \end{pmatrix}. \tag{4}$$

We use the estimated utility function to derive a weighting function such that

$$\mathbf{W}_a(\mathbf{\Phi}_a) \leftarrow \frac{e^{\kappa \tilde{\mathbf{u}}_a(\mathbf{\Phi}_a)}}{\|e^{\kappa \tilde{\mathbf{u}}_a(\mathbf{\Phi}_a)}\|}, \tag{5}$$

where κ is a selectivity parameter, so larger κ gives the more preference is given to larger estimated utility values. We use the weighting function to preferentially sample from the possible parameters $\mathbf{\Phi}_a$.

We propose using observations about the system's performance to optimize $\mathbf{W}_a(\mathbf{\Phi}_a)$ in a real-time manner. To do this, the attacker uses the observed utility $u_a^{obs}(k)$ obtained from the random sample $\boldsymbol{\phi}_a^k$ obtained with the weighting function $\mathbf{W}_a(\mathbf{\Phi}_a)$ to update its estimated utility vector as

$$\tilde{u}_a(\boldsymbol{\phi}_a) \leftarrow \lambda \tilde{u}_a(\boldsymbol{\phi}_a) + \beta u_a^{obs}(k). \tag{6}$$

In this algorithm the constant $\beta \geq 0$ is used as a weighting factor to avoid a high variance utility from forcing a rapid change in the probability distribution. The constant $\lambda \geq 0$ is used as a forgetting factor to avoid past history from causing

Data: $\mathbf{\Phi}_a^r$, $\mathbf{u_a^{obs}}$, $\mathbf{v}$, $\mathbf{b}$
Result: Trained neural network approximator variables $\beta_i, \forall i = 1, ..., m$
$\mathbf{E} \leftarrow \mathbf{u_a^{obs}}$
for *i=1:m* **do**
  $\mathbf{g} \leftarrow \sigma(v_i \mathbf{\Phi}_a^r + b_i)$
  $\beta_i \leftarrow \frac{\mathbf{E}^T \mathbf{g}}{\|\mathbf{g}\|^2}$
  $\mathbf{E} \leftarrow \mathbf{E} - \beta_i g$
end

algorithm 2: Universal approximator training

bad results in the transient case. The attacker then uses the new estimated utility to recalculate $\mathbf{W}_a(\mathbf{\Phi}_a)$ which it uses for random sampling in the next round. We initialize the estimated utility to the ones vector $\tilde{\mathbf{u}}_a(\mathbf{\Phi}_a) = \mathbf{1}$. We summarize this equation in Algorithm 1 and evaluate it in Section 5.

3.2. Universal Approximation

We next introduce the universal approximator algorithm using neural network [18]. Neural networks, inspired by neurological biology, feeds an input into many activator functions which are set to one if the input is over a threshold and set to negative one otherwise. The outputs of the activator functions are then multiplied by weights and summed. Suppose we desire to approximate a function $y = f(x)$ over some range using a neural network with randomly selected activator function thresholds. To do this we randomly sample over $y = f(x)$ and use these values to train the weights for our neural network. Once the neural network has been trained, the output corresponding to any input in the range is estimated by putting the input into the neural network and observing its output.

Applying the universal neural networks approach to this work, we randomly select a small set of random parameters. We use these parameters to perturb the system, and the corresponding utilities are measured. We then use the parameter inputs and utility outputs to train a universal neural network approximator. The approximation of the systems performance is used to find the minimal utility function.

To implement this, we uniformly randomly choose a parameter set for j time steps. Thus we end up with a vector of parameter vectors $\mathbf{\Phi}_a^r = [\boldsymbol{\phi}_a^1, \boldsymbol{\phi}_a^2, ..., \boldsymbol{\phi}_a^j]$ and a vector of observations $\mathbf{u_a^{obs}} = [u_a^{obs}(1), u_a^{obs}(2), ..., u_a^{obs}(j)]$. To train the neural network with this data, we select two sets of m random variables $\boldsymbol{b} = [b_1, b_2, ..., b_m]$ and $\boldsymbol{v} = [v_1, v_2, ..., v_m]$, where $v_i \in \{v_{min}, v_{max}\}$ and $b_i \in \{b_{min}, b_{max}\}$, $\forall i = 1, 2, ..., m$. We then define the output of our function as

$$\tilde{u}_a(\boldsymbol{\phi}_a) = \sum_{i=1}^{m} \beta_i \sigma(v_i \boldsymbol{\phi}_a + b_i) \tag{7}$$

$\tilde{\mathbf{u}}_a(\mathbf{\Phi}_a, \mathbf{\Phi}_d) \leftarrow \mathbf{1}$
$\tilde{\mathbf{W}}_d(\mathbf{\Phi}_d) \leftarrow \mathbf{1}$
for $k = 1 : \infty$ **do**
  $\mathbf{W}_a(\mathbf{\Phi}_a) \leftarrow \frac{e^{\kappa \tilde{\mathbf{u}}_a(\mathbf{\Phi}_a, \mathbf{\Phi}_d)} \tilde{\mathbf{W}}_d(\mathbf{\Phi}_d)}{\| e^{\kappa \tilde{\mathbf{u}}_a(\mathbf{\Phi}_a, \mathbf{\Phi}_d)} \tilde{\mathbf{W}}_d(\mathbf{\Phi}_d) \|}$
  $\boldsymbol{\phi}_a^k \leftarrow \text{rand\_sample}(\mathbf{\Phi}_a, \mathbf{W}_a(\mathbf{\Phi}_a))$
  $\tilde{\mathbf{u}}_{\mathbf{a}}(\boldsymbol{\phi}_a^k, \boldsymbol{\phi}_d^k) = \lambda \tilde{\mathbf{u}}_{\mathbf{a}}(\boldsymbol{\phi}_a^k, \boldsymbol{\phi}_d^k) + \beta u_a^{obs}(k)$
  $\tilde{\mathbf{W}}_d(\boldsymbol{\phi}_d^k) \leftarrow \alpha + \tilde{\mathbf{W}}_d(\boldsymbol{\phi}_d^k)$
  $\tilde{\mathbf{W}}_d(\mathbf{\Phi}_d) \leftarrow \mu \tilde{\mathbf{W}}_d(\mathbf{\Phi}_d)$
end

algorithm 3: Two-Player Weighted Observation Algorithm

Where

$$\sigma(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ -1 & \text{if } z < 0. \end{cases} \tag{8}$$

We train the β values by setting them one by one using the algorithm 2. We then use these values to approximate for any given ϕ_a using (7). This allows us to choose $\phi^\star$ which is the solution to the problem

$$\phi^\star = \quad \arg\max_{\phi} \quad \tilde{u}_a(\phi). \tag{9}$$

To account for the case where the defender is transient, we use a random reset function. When the reset function is triggered we use a random strategies for j time steps and use the j observations to re-train the neural network. We could trigger the reset periodically or when the observed utility function is sufficiently under the estimated utility.

4. Two Adaptive Player Formulation

In this section, we discuss the formulation of an observe-and-adapt strategy for the case where both players are adaptive or random. This is accomplished using a weighted observation algorithm similar to that used in the fixed and transient cases discussed previously. Similar to the one-player weighted observation algorithm, we discuss this from the role of the attacker only, and the results of the interaction of the two is analyzed in the Section 5.

We design an attacker that considers its utility with respect to the defender's and attacker's parameters. Thus we define the estimated utility function as

$$\tilde{\mathbf{u}}_a(\mathbf{\Phi}_a, \mathbf{\Phi}_d) = \begin{pmatrix} \tilde{u}_a(\boldsymbol{\phi}_a^1, \boldsymbol{\phi}_d^1) & \tilde{u}_a(\boldsymbol{\phi}_a^1, \boldsymbol{\phi}_d^2) & \dots & \tilde{u}_a(\boldsymbol{\phi}_a^1, \boldsymbol{\phi}_d^h) \\ \tilde{u}_a(\boldsymbol{\phi}_a^2, \boldsymbol{\phi}_d^1) & \tilde{u}_a(\boldsymbol{\phi}_a^2, \boldsymbol{\phi}_d^2) & \dots & \tilde{u}_a(\boldsymbol{\phi}_a^1, \boldsymbol{\phi}_d^h) \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{u}_a(\boldsymbol{\phi}_a^m, \boldsymbol{\phi}_d^1) & \tilde{u}_a(\boldsymbol{\phi}_a^m, \boldsymbol{\phi}_d^2) & \dots & \tilde{u}_a(\boldsymbol{\phi}_a^m, \boldsymbol{\phi}_d^h) \end{pmatrix}. \tag{10}$$

This function allows for an attacker to estimate how well it does for any combination of attacker and defender parameter vector. However, the attacker does

not know before a round what the defender's strategy will be. Hence, we define a matrix to give relative weights to the likelihood a defender chooses particular parameters. This weighting vector depends only on the defenders parameters so we define it as

$$\tilde{\mathbf{W}}_d(\mathbf{\Phi}_d) = \begin{pmatrix} \tilde{W}_d(\boldsymbol{\phi}_d^1) \\ \tilde{W}_d(\boldsymbol{\phi}_d^2) \\ \vdots \\ \tilde{W}_d(\boldsymbol{\phi}_d^h) \end{pmatrix}. \tag{11}$$

Using $\tilde{\mathbf{u}}_a(\mathbf{\Phi}_a, \mathbf{\Phi}_d)$ and $\tilde{\mathbf{W}}_d(\mathbf{\Phi}_d)$ we define the attacker's weighting function as

$$\mathbf{W}_a(\mathbf{\Phi}_a) \leftarrow \frac{e^{\kappa \tilde{\mathbf{u}}_a(\mathbf{\Phi}_a, \mathbf{\Phi}_d)} \tilde{\mathbf{W}}_d(\mathbf{\Phi}_d)}{\| e^{\kappa \tilde{\mathbf{u}}_a(\mathbf{\Phi}_a, \mathbf{\Phi}_d)} \tilde{\mathbf{W}}_d(\mathbf{\Phi}_d) \|}, \tag{12}$$

where $\|\cdot\|$ is the Eucledian norm and κ is the selectivity factor as used previously, that is tuned to balance learning versus maximizing utility.

We again propose using observations about the system performance and defender's parameter choice to update the attacker's utility estimate function. We define the update equation as

$$\tilde{\mathbf{u}}_\mathbf{a}(\boldsymbol{\phi}_a^k, \boldsymbol{\phi}_d^k) \leftarrow \lambda \tilde{\mathbf{u}}_\mathbf{a}(\boldsymbol{\phi}_a^k, \boldsymbol{\phi}_d^k) + \beta u_a^{obs}(k), \tag{13}$$

where β is a weighting factor to avoid a high variance utility and λ is a forgetting factor as described previously.

We propose using observations about the defender's choice in parameters to update the matrix $\tilde{\mathbf{W}}_d(\mathbf{\Phi}_d)$. We do this in two steps, the first increases the weight of whatever parameters the defender last played by a constant α such that

$$\tilde{\mathbf{W}}_d(\boldsymbol{\phi}_d^k) \leftarrow \alpha + \tilde{\mathbf{W}}_d(\boldsymbol{\phi}_d^k). \tag{14}$$

The attacker then updates all the values in the matrix using a scaling factor $\mu < 1$ which decreases all the values. Thus our second update operation is

$$\tilde{\mathbf{W}}_d(\mathbf{\Phi}_d) \leftarrow \mu \tilde{\mathbf{W}}_d(\mathbf{\Phi}_d). \tag{15}$$

To start the algorithm we choose to use an initialization to $\tilde{\mathbf{u}}_a(\mathbf{\Phi}_a, \mathbf{\Phi}_d) \leftarrow \mathbf{1}$ and $\tilde{\mathbf{W}}_d(\mathbf{\Phi}_d) \leftarrow \mathbf{1}$. This algorithm is summarized in Algorithm 3 and evaluated in Section 5.

4.1. Cross-Layer Design

Our initial presentation allows for the freedom to choose the utility as any measurable device characteristic or combination of measurable characteristics. This allows for intuitive design across layers since utility is not confined to one layer.

Likewise, the use of a model with generic parameters that map to utility allow for selection of parameters and strategies from many places in the stack. The main caveat here is that the set of parameters selected must be able to be searched. One of the main difficulties of using this type of approach over many layers with a numerous parameter is the search space becomes intractable.

5. Simulation

We design and simulate two multi-round games as a proof-of-concept for using observe-and-adapt algorithms for adaptive security. The first game simulates a generic jamming game where both players choose their power level and the outcome of the game is based on a generic utility mapping with noise. The second game focuses on a jamming scenario simulating 802.11b in which both players control their power levels. The utility in this game is a mix of the PER rate from a simulation of the jamming scenario and the players power usage.

We explore multiple two-player scenarios including an adaptive player and fixed player, an adaptive player and transient player, an adaptive player and random player, and two adaptive players. We assume that the adaptive player is always rational and aims to maximize its utility by increasing its impact while minimizing its power usage.

In both of these games the players select the strategy a priori. An adaptive player aims to adapt one parameter, power level over a time, to maximize its utility. We define p_a as the attacker's power and p_d as the defender's power. For both games the players choose power levels from a discrete subset of choices.

A fixed player randomly chooses its power level prior to playing the game. A transient player chooses a new power level at random every 600 rounds. A random player selects a new power level randomly every round. An adaptive player uses the strategies described in Section 3 and 4.

5.1. Generic Jamming Game

The generic jamming game is setup using a cost matrix that is designed to emulate possible costs of jamming. The game is not zero sum, but rather our utility map for the attacker is defined as

$$u_d(p_a, p_d) = u_v + u_E \tag{16}$$

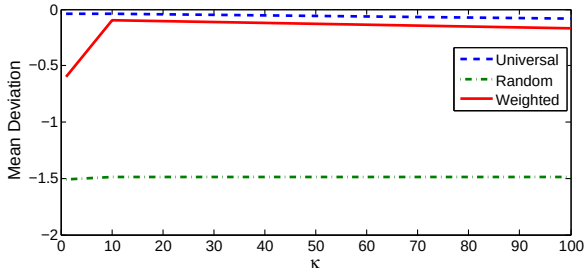
where u_v is the utility from system degradation and u_E is the utility from the energy expenditure. This leads us to

$$u_a(p_a, p_d) = \begin{pmatrix} 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & -1 \\ -2 & 2 & 2 & -2 \\ -3 & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 4 & 4 & 4 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 \\ -1 & -1 & -1 & -1 \\ -2 & -2 & -2 & -2 \\ -3 & -3 & -3 & -3 \end{pmatrix}. \tag{17}$$

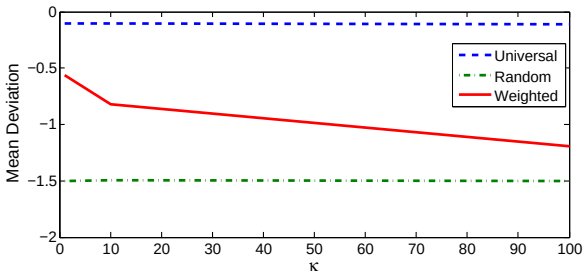
The function for u_v is 4 units whenever the jammer successfully jams the channel. The utility for u_E is minus one unit for every power level the jammer uses.

We likewise define the defenders utility matrix as

$$u_d(p_a, p_d) = \begin{pmatrix} -1 & 6 & 4 & 1 \\ 1 & -1 & 4 & 1 \\ 1 & -1 & -2 & 1 \\ 1 & -1 & -2 & -3 \end{pmatrix}. \tag{18}$$



(a) Generic jamming game, fixed Defender



(b) Generic Jamming game, transient Defender

Figure 3: In 3(a) we show the mean deviation from 20 plays of 500 rounds of the fixed game and in 3(b) we show the mean deviation from 20 plays of 10000 rounds of the transient game. We show results for an attacker using weighted observation, universal approximators, and a random strategy. The choice of the selectivity constant κ is varied along the x-axis.

This equation is similar but awards utility when the attacker is off and the jammer is on.

For the weighted observation algorithm we set $\lambda = .9$, $\beta = .1$, $\alpha = 1$, and $\mu = .9$. For the universal approximator algorithm we select $v_{min} = -.2$, $v_{max} = .2$, $b_{min} = -2$, $b_{max} = 2$, the number of nodes $m = 1000$, and the number of samples $j = 12$. For the near static universal approximator, we reset every 300 rounds.

5.1.1. Results

Fixed - For the fixed defender we show results from an attacker using the weighted observation and universal approximation algorithm. One statistic that we use to analyze the algorithms performance is the deviation from maximum efficiency. We define deviation from maximum efficiency as the difference between the observed utility and the expectation of the utility of the optimal strategy. An ideal attacker would expect their deviation from maximum to be the variance caused from noise in the game. We define the mean deviation from maximum as the mean of the deviation from maximum for all rounds of a game. The mean deviation from an ideal attacker would be zero. In Figure 3(a) we show the mean of the mean deviation from 20 games of 500 rounds for each algorithm for various κ values. We show that an attacker using a random strategy achieves a mean deviation of -1.5. We also show when an attacker uses the

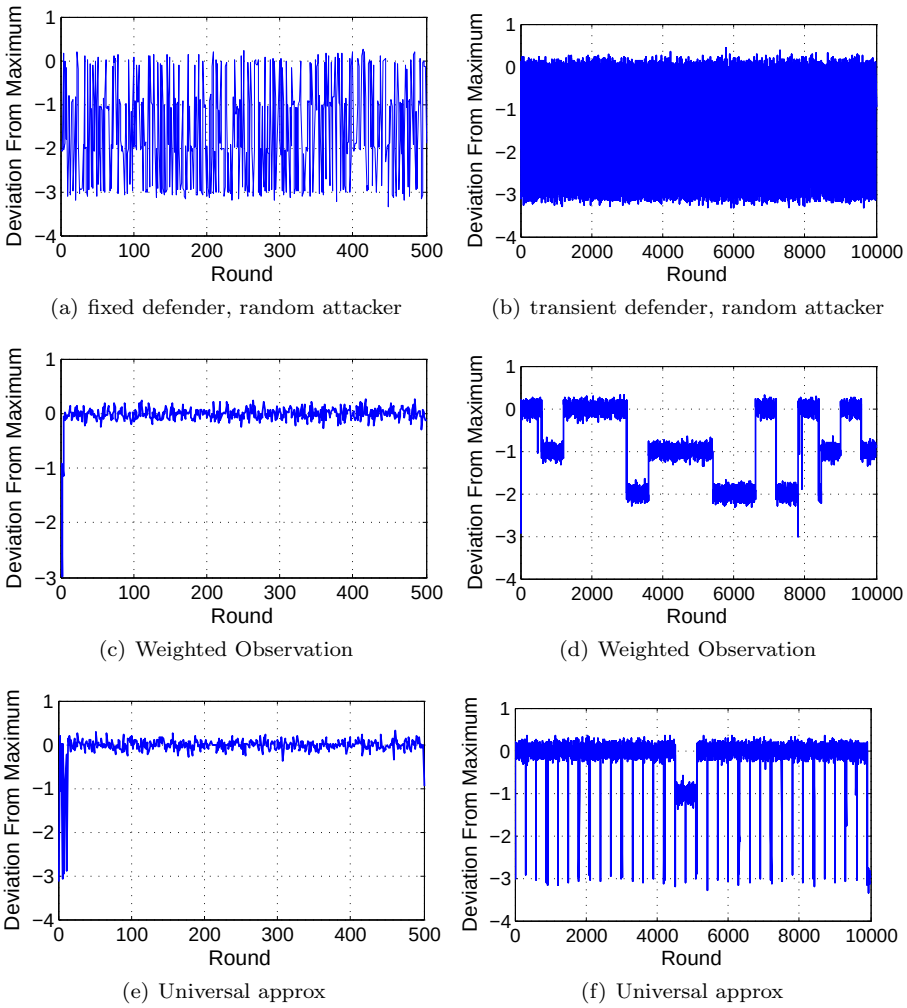
weighted observation algorithm it achieves a mean deviation over -.2 and using the universal approximator the mean deviation is over -.1. The importance of the significance value κ is highlighted in this figure. As expected the higher κ value increases selectivity, choosing the optimal value more and limiting exploration of the search space.

We show runs of our algorithms in Figure 4 for the fixed game. In Figure 4(a) we show the results for the random attacker, the high variation in the deviation from maximum in this plot confirms the sub-optimality of a random strategy. In Figure 4(c), we show the results for the weighted observation algorithm with $\kappa = 10$, which after a short time obtains ideal results. Lastly, in Figure 4(e) we show the results for the universal approximator algorithm, which after the 12 round perturbation obtains ideal results.

Transient - For the transient defender we again consider an attacker using a random algorithm, a weighted observation algorithm ,and a universal approximator algorithm. The transient defender chooses a strategy randomly every 600 rounds. The attacker using the universal approximator algorithm randomly samples 12 data points every 300 rounds and uses these values to re-train its neural network. In Figure 3(a) we show the mean of the mean deviation from 20 games of 10000 rounds for each algorithm for various κ values. The figure shows that the attacker using the universal approximator algorithm again achieves a mean deviation of over -.2, which is much better than the -1.5 from the random algorithm. The attacker using The weighted observer algorithm shows varied results, though always better than random. The higher the selectivity value, κ the worse the system performs, which is to be expected. This result is the opposite of the fixed-case, where high selectivity is beneficial. This is easy to explain by considering the selectivity discourages trying new values and adapting the system when the transient system changes.

We show runs of the attacker using adaptive algorithms against a transient defender in Figure 4. In Figure 4(f) we show a run of an attacker using the universal approximator algorithm. The algorithm continually performs near ideally with few exceptions. The one discrepancy at around 4500 rounds is likely caused by a bad selection of random variables for the $v_i's$ and $b_i's$ in the neural network. The results for an attacker using the weighted observer algorithm are shown in Figure 4(d). These results are less consistent, but on average better than random.

Adaptive - In Figure 5, we show the results of our weighted approximation algorithm for 3 cases and various κ values. We again use the mean utility of 20 games with 10000 rounds. The three cases we use are two adaptive players, two random players, and a random player and an adaptive player. The results when only the defender adapts shows the defender's utility improve and the attacker's utility remains unchanged. When only the attacker adapts the attacker's utility improves and the defender's utility improves. This is likely due to the attacker's adaptation causing the attacker to choose power levels that are, on average, better for the attacker. The attacker does not see the same gain because the expectation of his random strategy is zero for any power level choice of the defender.



(a) fixed defender, random attacker

(b) transient defender, random attacker

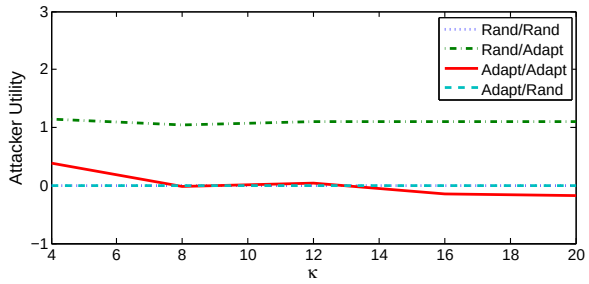
(c) Weighted Observation

(d) Weighted Observation

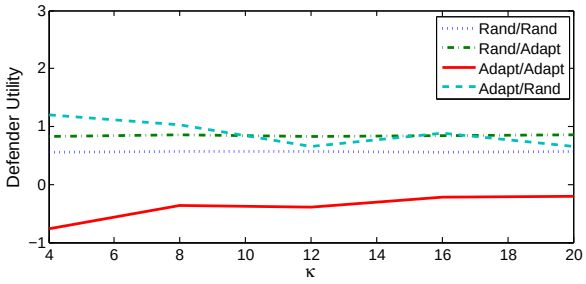
(e) Universal approx

(f) Universal approx

Figure 4: In this figure, we show the attacker's deviation from maximum for a fixed defender and transient defender using the generic jamming game. We show the performance of a random attacker, an attacker using the weighted observation algorithm and an attacker using the universal approximator algorithm.



(a) Attacker Utility



(b) Defender Utility

Figure 5: In this figure, we show the average utility of two players playing the generic jamming game. We look at the performance of three two-player games: two random players, a random player and an adaptive player, and two adaptive players. The legend in the graph is read as: (Defender strategy)/(Attacker strategy).

The results for both players adapting is also interesting. If both players use a selectivity of $\kappa = 4$ the attacker has a slight gain, the defender has a loss of almost one. The defender consistently has worse performance when both players adapt. Given this information the use of adaptation by the individual players can be decided on the fly. For example, a defender who is getting performance worse than random could choose to use a random strategy after a number of adaptive trials.

5.2. 802.11b Jamming Game

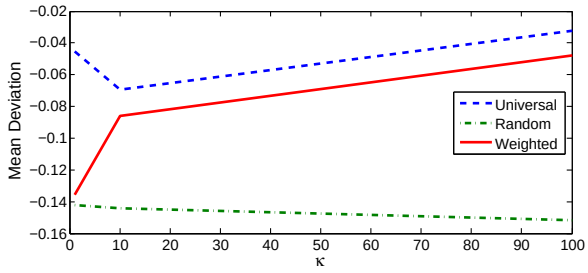
Our second sample system uses the IEEE 802.11b model [19] distributed with the Matlab Simulink Communications Blockset. We add a periodic jammer into this model, a PER measurement system, and control for the jammer's power and transmitter's power which we use for p_a and p_d respectively. The 802.11b jamming power game is very similar to the one discussed above, except in the 802.11b domain. The defender and attacker can choose power levels in the range of 0 and 10. With 100 rounds of the game for each combination of parameters we can summarize this with a mean of

$$\mu_{PER}(p_a, p_d) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0.73 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0.95 & 0.29 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0.91 & 0.69 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0.94 & 0.73 & 0.24 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0.98 & 0.92 & 0.73 & 0.28 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0.98 & 0.93 & 0.76 & 0.29 & 0.25 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0.98 & 0.90 & 0.68 & 0.68 & 0.23 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0.98 & 0.94 & 0.92 & 0.72 & 0.41 & 0.24 & 0 & 0 & 0 & 0 \\ 1 & 0.98 & 0.88 & 0.91 & 0.68 & 0.66 & 0.23 & 0.26 & 0 & 0 & 0 \end{pmatrix} \tag{19}$$

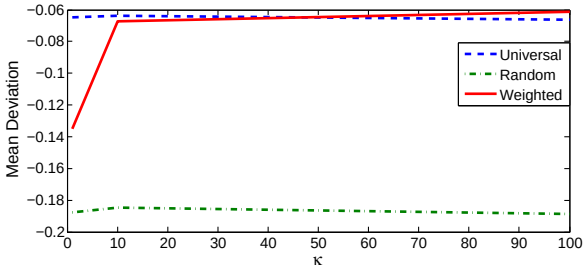
and standard deviation of

$$\sigma_{PER}(p_a, p_d) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.28 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.11 & 0.24 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.17 & 0.30 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.13 & 0.27 & 0.25 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.16 & 0.30 & 0.24 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.13 & 0.25 & 0.24 & 0.25 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.18 & 0.27 & 0.27 & 0.25 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.13 & 0.15 & 0.30 & 0.32 & 0.25 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.19 & 0.17 & 0.30 & 0.27 & 0.24 & 0.24 & 0 & 0 & 0 \end{pmatrix}. \tag{20}$$

Utility Functions - In this game the dependability metric is defined as $v = 1 - \text{PER}$. The energy for the defender is calculated as $E_d^k = p_d^2$ and $E_{d,max} = 100$.



(a) fixed defender



(b) transient defender

Figure 6: In this figure, we show an adaptive attacker's gain in utility over a random strategy for a fixed and transient 802.11b jamming game. We consider the use of both universal approximators and weighted observations.

The energy for the defender is calculated as $E_a^k = p_d^2$ and $E_{a,max} = 100$. The weights for the utility functions are set as $w_{E,d} = w_{E,a} = .4$ and $w_{v,d} = w_{v,a} = .6$. Using (2) and (3) the utility for the attacker becomes

$$u_a^k = .6\text{PER} + .4\left(1 - \frac{p_a^2}{100}\right) \tag{21}$$

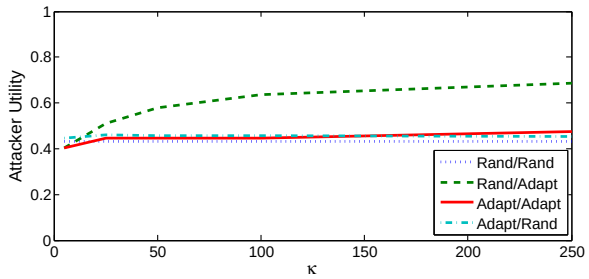
and the utility for the defender is

$$u_d^k = .6(1 - \text{PER}) + .4\left(1 - \frac{p_d^2}{100}\right). \tag{22}$$

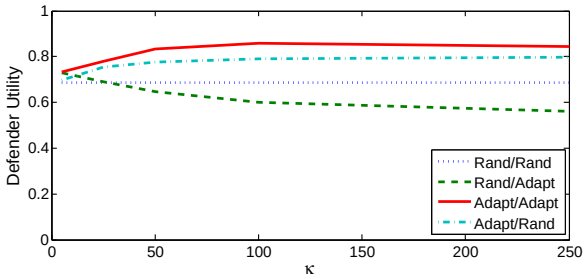
5.2.1. Results

Fixed and Transient - We again test the weighted observation algorithm and universal approximator algorithm for the fixed and transient game, the results are shown in Figure 6. When the attacker adapts it clearly out-performs a random attacker. The weighted observation method, with a sufficiently high κ value, performs as well as the universal approximator. It is interesting to note in this game that the higher selectivity constant always increases the performance of the system. This is different then the generic case and also make an important distinction that the selection of κ depends on the utility function.

Adaptive - In Figure 7 we show the results for the interaction of adaptive and random players. We again average 20 games with 10000 rounds each. This



(a) Attacker Utility



(b) Defedner Utility

Figure 7: In this figure we show the results for the 802.11b jamming game for adaptive players. The legend is read as (Defender's strategy)/(Attacker's strategy).

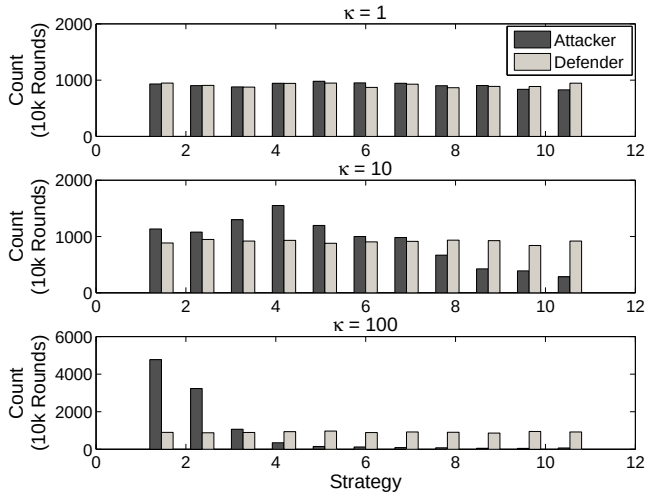
result is interesting for the attacker because it only make a large gain in it's performance when it adapts and the defender does not. In any other case, the attacker performs near the baseline case of both player being random. The defender on the other hand makes a gain in either case when it adapts. When it does not adapt and the attacker does it looses utility.

In Figure 8, we show some intuition about the importance of selectivity in the 802.11b jamming game. In this figure, we show the effect of varying selectivity values, κ, for the case of an adaptive player opposing a random player as well as two adaptive players competing. We can see that when the selectivity value is one, we get near random behavior in any case. When the value increases to 10, we see higher selection and when it is 100, we see clear favorite strategies. It is interesting to note that the strategies favored in the $\kappa = 10$ and $\kappa = 100$ cases are not the same.

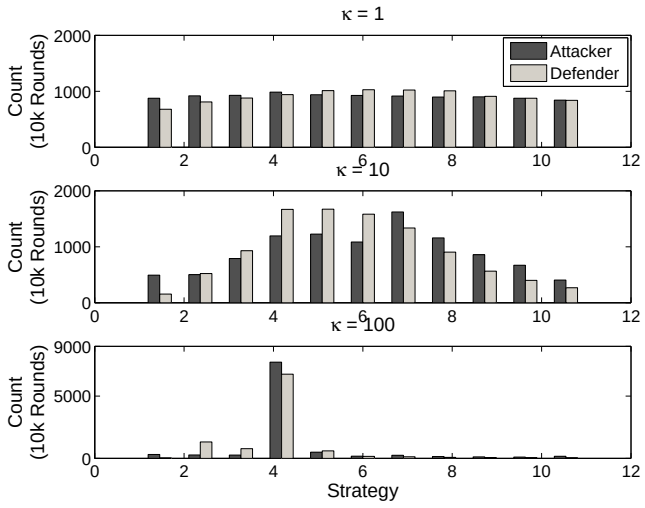
6. Related Works

In this section, we highlight similar work in the areas of adaptive security consider recent developments of secure communications as well game theoretic jamming analysis. For a detailed look at the historic development of the wider field of jamming see Pelechrinis et al. [2].

One adaptive strategy to mitigate jamming was proposed by Liu et al. [8], who suggest the SPREAD hopping scheme to make a system robust to cross



(a) Adaptive Attacker, Random Defender



(b) Adaptive Attacker and Defender

Figure 8: In this figure, we show the effect of the κ constant on the systems performance. This lends explanation to the effect of this as a selectivity constant, and shows why varying causes differences in performance. Particularly in the case of transient and adaptive opponents, the ability to unlearn is much more difficult with high κ values but they give the advantage of decreased testing of sub-optimal results.

layer denial of service attacks. This work suggests that a set of protocols at each layer of the stack are chosen and then synchronized nodes switch between protocols using a pseudo-random pattern. In our work, we provide a framework for escaping jamming attacks that is complimentary to this approach. A set of pseudo-random protocols could be chosen and then the observe-and-adapt approach could find optimal parameter values or determine when a less costly protocol could be chosen.

There has also been work in using current protocols or modified versions of current protocols to mitigate jamming threats. Wood et al. [13], have suggested a set of MAC layer modifications which mitigate jamming attacks in the 802.15.4 domain. Likewise, Pelechrinis et al. [12], have suggested tuning rate adaptation and carrier sensing threshold in 802.11 to achieve resiliency against jammers. We build on this line of working, showing that adaptation and appropriate tuning of parameters increases the robustness of a system and provide a general framework for studying this problem.

Designs for adaptive jamming attacks have been considered at both the MAC and PHY layer. Richa et al. [6], propose using an adaptive attacker that aims at the MAC layer. This paper uses feedback from previous rounds and can use all this history to optimize their current attack decision. DeBruhl et al. [5], have suggested an adaptive periodic jamming attack that uses observed information from the system to optimize its duty cycle and power parameters. Our current work is a more general framework which can be extended to use both MAC layer attack optimization and PHY layer attack optimization, building on both previous works.

Research in offline optimization of a jamming attack is also an open field. Li et al. [4], have shown the design of an optimal jammer which tries to maximize impact without being detected. This work presents a design based on the jamming probability, transmission range and network access probability which can provide for optimal solutions to when to jam. We again work in the spirit of this paper and attempt to optimize our attack to have maximum impact while minimizing cost, and extend it by considering how such an attacker would work against an defender which is also trying to adapt. Our paper approaches this differently in assuming that adaptation is online and has to use information from the system.

There has also been wide consideration of using a game-theoretic solution to optimize problems in the jamming and anti-jamming domain [14, 15, 20, 21, 22]. Thamilarasu and Sridhar [15], have shown optimal attack and detection design for a jamming game. The optimal detection balances a cross layer detection strategy and energy usage. Pelechrinis et al. [14], use a game-theoretic framework to analyze frequency hopping and show the effect of orthogonal channels on its effectiveness. Zhu et al. [20] consider an attacker that balances compromising security via eavesdropping and availability via jamming. Firouzbakht et al. [21] consider a system where the attacker can choose various power levels to transmit at and the defender can choose various coding rates. These works all consider perfect knowledge of the system to derive a rational strategy in real time. Our work looks to compliment these previous works by exploring

an online adaptive system that does not assume perfect knowledge nor a priori optimization. For a survey of the wider use of game theory in this field see Manshaei et al. [23].

It is also useful to consider the cognitive radio domain when exploring adaptation in communication. This domain is both beneficial to support the use of artificial intelligence approaches [16] as well as to provide direct exploration of robust adaptive communications. Brown et al. provide a summary of the new attacks that are opened up do to the unique nature of cognitive radio [24]. Wang et al. [25] propose using cognitive radio to work around jammed spectrum regions. This work assumes that attacker stays in a region after a sensing period, in practice attackers have no incentive to be this cooperative. Our work is complimented by advances in CRN learning and adaptation techniques as they increase the options a defender has to avoid jamming.

7. Conclusion

In this work, we consider the interaction of adaptive defenders and attackers. To do this we provide a framework that allows for both an attacker and defender to observe the system and adapt their strategy based on observations. We show that in the case of an adaptive player competing against a static or random opponent that they are able to improve their impact or decrease their cost. In the case that both players adapt we show that their utility is very suggestive of the bias in the game that they are playing. In the future this framework can be expanded to include more adaptive strategies to allow deeper exploration of the field. The framework can also be analyzed using formal methods to determine what type of guarantees and properties can be derived. This could include a game theoretic analysis as well as a more robust simulation based analysis.

References

[1] D. J. Torrieri, Principles of Secure Communication Systems, 2nd ed., Artech House, Boston, 1992.

[2] K. Pelechrinis, M. Iliofotou, S. Krishnamurthy, Denial-of-service attacks in wireless networks: the case of jammers, IEEE Comm Surveys and Tutorials (2011).

[3] M. Čagalj, S. Ganeriwal, I. Aad, J.-P. Hubaux, On cheating in CSMA/CA ad hoc networks, in: Proc. 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'05), Miami, FL, USA, 2005.

[4] M. Li, I. Koutsopoulos, R. Poovendran, Optimal jamming attacks and network defense policies in wireless sensor networks, in: 26th IEEE Conference on Computer Communication (InfoCom'07), 2007.

[5] B. DeBruhl, Y. Kim, Z. Weinberg, P. Tague, Stir-ing the wireless ether with self-tuned, inference-based, real-time jamming, in: Proc. 9th IEEE International Conference on Mobile and Ad Hoc and Sensor Systems, Las Vegas, USA, 2012.

[6] B. Awerbuch, A. Richa, C. Scheideler, A jamming-resistant mac protocol for single-hop wireless networks, in: Proc. of the 27th ACM symposium on Principles of distributed computing, Toronto, Canada, 2008.

[7] M. Wilhelm, I. Martinovic, J. Schmitt, V. Lenders, Reactive jamming in wireless networks: How realistic is the threat?, in: Proc. 4th ACM Conference on Wireless Network Security, Hamburg, Germany, 2011.

[8] X. Liu, G. Noubir, R. Sundaram, S. Tan, SPREAD: Foiling smart jammers using multi-layer agility, in: 26th IEEE International Conference on Computer Communications (INFOCOM'07), Anchorage, AK, USA, 2007.

[9] D. Slater, R. Poovendran, P. Tague, B. J. Matt, Tradeoffs between jamming resilience and communication efficiency in key establishment, ACM Mobile Computing and Communication Review 13 (2009).

[10] B. DeBruhl, P. Tague, Adaptive filtering techniques for jamming mitigation, in: 2nd International Conference on Pervasive and Embedded Computering and Communication Systems (PECCS'12), 2012.

[11] R. Liu, R. Ying, Anti-jamming filtering in the autocorrelation domain, IEEE Signal Processing Letters 11 (2004) 525–528.

[12] K. Pelechrinis, I. Broustis, S. V. Krishnamurthy, C. Gkantsidis, Ares: an anti-jamming reinforcement system for 802.11 networks, in: Proceedings of the 5th international conference on Emerging networking experiments and technologies, 2009.

[13] A. D. Wood, J. A. Stankovic, G. Zhou, DEEJAM: Defeating energy-efficient jamming in IEEE 802.15.4-based wireless networks, in: Proc. 4th Annual IEEE Conference on Sensor, Mesh and Ad Hoc Communication Networks (SECON'07), San Diego, CA, USA, 2007.

[14] K. Pelechrinis, C. Koufogiannakis, S. V. Krishnamurthy, Gaming the jammer: Is frequency hopping effective?, in: Proc. 7th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt'09), Seoul, Korea, 2009.

[15] G. Thamilarasu, R. Sridhar, Game theoretic modeling of jamming attacks in ad hoc networks, in: Proc. International Conference on Computer Communications and Networks (ICCCN'09), San Fransico, CA, USA, 2009.

[16] K. Haigh, C. Partridge, Can artificial intelligence meet the cognitive networking challenge?, Dayton, OH, Sep (2011).

[17] R. T. Marler, J. S. Arora, Survey of multi-objective optimization methods for engineering, Structural and Multidisciplinary Optimization 26 (2004) 369–395.

[18] G. Huang, Q. Zhu, C. Siew, Extreme learning machine: Theory and applications, Neurocomputing 70 (2006).

[19] Simulink: Ieee 802.11b wlan physical layer, 2012. `http://www.mathworks.com/help/comm/examples/ieee-802-11b-wlan-physical-layer.html`.

[20] Q. Zhu, W. Saad, Z. Han, H. Poor, T. Basar, Eavesdropping and jamming in next-generation wireless networks: A game-theoretic approach, in: Military Communications Conference, IEEE, 2011, pp. 119–124.

[21] K. Firouzbakht, G. Noubir, M. Salehi, 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, ACM, 2012, pp. 3–14.

[22] S. Bhattacharya, A. Khanafer, T. Basar, Switching behavior in optimal communication strategies for team jamming games under resource constraints, in: Control Applications (CCA), 2011 IEEE International Conference on, IEEE, 2011, pp. 1232–1237.

[23] M. Manshaei, Q. Zhu, T. Alpcan, T. Basar, J. Hubaux, Game theory meets network security and privacy, ACM transaction on Computational Logic 5 (2011).

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

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