Allocating Security Analysts to Cyber Alerts Using Markov Games

Noah Dunstatter
Department of Computer Science
Texas State University
San Marcos, Texas 78666
Email: nfd8@txstate.edu

Mina Guirguis
Department of Computer Science
Texas State University
San Marcos, Texas 78666
Email: msg@txstate.edu

Alireza Tahsini
Department of Computer Science
Texas State University
San Marcos, Texas 78666
Email: a_t325@txstate.edu

***Abstract*—Allocating cyber-security analysts to incoming cyber alerts is an important task in any organization employing cyber-defense mechanisms. Alerts are typically generated when intrusion detection software on computer systems (e.g., servers, routers) detect abnormal or suspicious activity. Based on the respective significance level of the alerts, some are assigned to cyber-security analysts for further investigation. Due to the wide range of potential attacks coupled with high degrees of attack sophistication, identifying what constitutes a true attack is a challenging problem, especially for organizations performing critical operations (e.g., military bases, financial institutions, etc.) that are constantly being subjected to cyber attacks every day. In this paper, we develop a game-theoretical framework that assigns cyber-security analysts to cyber alerts to minimize the overall risk faced by an organization. Our approach considers a series of games between the attacker and the defender in which a state is maintained between sub-games. The state captures the availability of analysts as well as an attack budget metric that enables us to model the level of risk an attacker is willing to undertake. Through dynamic programming and Q-maximin value iteration-based algorithms, we identify optimal allocation strategies that take into account the current availability of analysts, the risk faced by the attacker, the incoming alerts, and the future outlook of the system. We assess the effectiveness of our allocation strategies by comparing them to other sensible heuristics (e.g., random, greedy and myopic). Our results show that our approach outperforms these other strategies in minimizing risk.**

I. INTRODUCTION

Motivation and Scope: The current state of cyber-defense is dire [8]. On one hand, new trends of more sophisticated, economically-driven, state-sponsored cyber attacks are on the rise; evidenced by the recent security breaches at both public and private institutions (e.g., the Pentagon [2], Sony [4], Target [14]). On the other hand, the emergence (and deployment) of Cyber-Physical Systems have enabled cyber attacks to cross from the cyber realm into our physical infrastructure, making them appealing vehicles for terrorism and terrorism-related activities [5]. Various organizations – in both the public and private sectors – are constantly being subjected to attacks that seek to disable, disrupt and/or breach their cyber infrastructure, especially during times of critical operations (e.g., before missions, political statements, software releases, etc.). The authors in [8] report an average of 17,000 alerts every week at surveyed organizations. Of these, roughly 19% (3,218) are estimated to be legitimate alerts, with only 4% (705) eventually being investigated.

One critical component of any institution's cyber-defense infrastructure is maintaining the appropriate workforce of cyber-security analysts who handle the investigations of incoming alerts. When an attack is launched against an organization, sensors (e.g., Intrusion Detection Systems, Anti-malware tools, etc.) deployed on various systems or machines (e.g., computers, servers, routers, etc.) typically generate cyber alerts with varying levels of severity. At the same time legitimate network activity can generate false-positive alerts. This typically high volume of false positives can drown out relevant alerts, as seen in the Target attack wherein malware alerts were repeatedly generated but not addressed [14]. To determine if an alert is a false positive it must first be assigned to a security analyst for investigation. Some of these alerts (e.g., those in a high risk level) have a higher probability of representing an ongoing attack and hence, must be prioritized when being assigned for investigation. As such, an allocation of cyber-security analysts to cyber alerts may depend on various factors, namely: (1) the expertise of the analysts, (2) the current availability of analysts, (3) the value of the machine from which the alerts originate, and (4) the associated threat level of the alerts.

Sophisticated adversaries can gain critical knowledge through probing attacks that observe the response time of the analysts in handling the attacks, and thus can choose the best attack method, along with the correct time to strike (e.g., when all the analysts are busy investigating other alerts). Unlike previous work that considered a one-shot game with a deterministic alert arrival [11], [12], in this paper we consider a Markov game model in which the defender and the adversary play a series of games with a *state* maintained between games. The stochastic nature of the Markov game is manifested in not knowing the exact numbers/levels of alerts that will arrive in the future, which impacts how allocations are made at the current time as well as the mixed strategy nature of the players' policies. We consider two variants of the problem, one with a finite horizon and one with an infinite horizon. In the finite horizon case, we develop optimal strategies via dynamic programming whereby optimal policies are not just based on the current state, but on the time that state is encountered. In the infinite horizon case, we use Q-maximin value iteration based approaches to obtain invariant optimal policies based on the state.

Contributions: In this paper, we develop a Markov game model that (1) captures the uncertainty in the alert arrival process, (2) accounts for analysts that spend more than one time step investigating an alert leading to *stateful* models,

(3) considers the adversarial nature of the network security domain from both attacker and defender perspectives, and (4) employs an attacker budget metric that captures their willingness to exposure. Through dynamic programming and Q-maximin value iteration approaches we obtain optimal worst-case allocation strategies that succeed in minimizing risk while still outperforming other allocation strategies such as random, greedy, and myopic.

Paper organization: In Section II, we put our work in context to other existing work. In Section III, we describe our models for both the finite and infinite horizon cases and present our solution methods for obtaining optimal worst-case policies. We present our evaluation in Section IV and conclude the paper with a summary of our contributions and future directions in Section V.

II. RELATED WORK

There has been some work that focused on scheduling and improving the efficiency of cyber-security analysts [1], [6], [7], [18]. The authors in [1] model the problem as a two-stage stochastic shift scheduling problem in which the first stage allocates the security experts and in the second stage, additional experts are allocated. The problem is discretized and solved using a column generation based heuristic. The authors in [6] study optimal allocation of alerts to analysts under a static workforce size with a fixed alert generation mechanism. In [7], the authors develop a reinforcement learning-based dynamic programming model to schedule cyber-security analysts, with the model based on a Markov Decision Process framework with stochastic load demands. In [18], the author describes different strategies for managing security incidents in a cyber-security operation center. The authors in [13] propose a queuing model to determine the readiness of a Cyber-Security Operations Center (CSOC). In our work, we depart from this previous research by explicitly considering the presence of a strategic adversary.

The use of game theory has been instrumental in advancing the state-of-the-art in security games and their wide range of applications [3], [15], [17], [9], [11], [12]. Some of the most recent and relevant lines of work involving analyst alert allocation are studied in [11] & [12]. Here the authors introduce a game-theoretic model to determine the best allocation of incoming cyber alerts to analysts. Their model, however, assumes a one shot-game in which both the alert resolution time and the arriving alert distribution are deterministically known.

III. MARKOV GAME MODEL

We consider a two-player zero-sum Markov game in which the Defender ($\mathcal{D}$) and the Attacker ($\mathcal{A}$) play a series of sub-games over some time horizon $\mathcal{T}$. At each time step $t \in \mathcal{T}$, a new batch of alerts $\theta \in \Theta$ arrives in which $\mathcal{A}$ chooses some alert level(s) to attack in and $\mathcal{D}$ attempts to detect and thwart the incoming attack(s) by assigning available analysts to the incoming alerts. Without loss of generality a time step may represent any period of elapsed time, however for the remainder of the paper we will assume a time step of one hour. We let $s \in \mathcal{S}$ denote the current state of the player resources (e.g., availability of analysts as well as the budget available to the attacker). We also let $\mathcal{D}_a$ and $\mathcal{A}_a$ denote the actions available to $\mathcal{D}$ and $\mathcal{A}$, respectively. We define a transition function $T : \mathcal{S} \times \mathcal{D}_a \times \mathcal{A}_a \rightarrow \Pi(\mathcal{S})$ which maps each state and player action pair to a probability distribution over possible next states. We let $T(s, a, d, s')$ denote the probability that, after taking actions $a \in \mathcal{A}_a$ and $d \in \mathcal{D}_a$ in state s, the system will make a transition into s'. In general, the system can be described as follows:

- **Alerts:** At every time step t, a batch of alerts $\theta \in \Theta$ arrives according to some probability distribution $\Pi(\Theta)$, where $\Theta = \{\theta_1, \theta_2, \ldots, \theta_{|\Theta|}\}$. Each alert $\sigma \in \theta$ belongs to one of three categories: High (h), Medium (m), or Low (l). Resolving an alert requires a certain number of time steps based on its category. The set holding each category's work-time (i.e., the number of time steps needed by an analyst to investigate and resolve an alert) is defined as $\mathcal{W} = \{w_h,\ w_m,\ w_l\}$ where $w_h > w_m > w_l$ and $w_h, w_m, w_l \in \mathbb{N}$. A similar reward structure $\mathcal{U} = \{u_h,\ u_m,\ u_l\}$, where $u_h > u_m > u_l$ and $u_h, u_m, u_l \in \mathbb{N}$, is defined for each category as well. If the alert σ^h is legitimate and is assigned to an analyst, $\mathcal{D}$ will receive a positive utility u_h. Whereas not assigning the legitimate alert results in a negative utility $-u_h$ for $\mathcal{D}$. If an alert is illegitimate (i.e., a false positive) then it awards no utility to $\mathcal{A}$ and $\mathcal{D}$, regardless of whether or not it is assigned. Since our model is zero-sum the corresponding utilities for $\mathcal{A}$ are simply the additive inverse of those awarded to $\mathcal{D}$. We assume that both players are aware of the set of possible alert batches Θ, as well as the respective probability of each batch. For example, a possible arrival set may be as follows: $\Theta = \{\theta_1,\ \theta_2\}$ where $\Pr(\theta_1)$ = 0.4 with θ_1 = $\{\sigma_1^h,\ \sigma_2^m,\ \sigma_3^m\}$, and $\Pr(\theta_2)$ = 0.6 with θ_2 = $\{\sigma_1^m,\ \sigma_2^l\}$. At the beginning of a particular game, both players are aware of the exact alert batch that has arrived (with future alert arrivals remaining probabilistic, which impacts the current resource allocations made by the players). It is important to note that not every alert may be assigned to an expert, and not every alert may represent a legitimate attack (i.e., alerts can be false positives).
- **The Defender:** $\mathcal{D}$ has n homogeneous cyber-security analysts available to handle incoming alerts. We define R_s as a vector of length n that describes the load of each analyst in state s. For example, $R_s = [0\ 2\ 1]$ means that $\mathcal{D}$ has 3 analysts on their team in which analyst 1 is free (has load 0), analyst 2 will be free after two time steps, and analyst 3 will be available after 1 time step. We also define the function $\mathcal{F}(R_s)$ as the number of analysts currently available for allocation in state s. In every time step t, $\mathcal{D}$ receives a batch of alerts θ and determines their allocation strategy based upon the current availability of analysts and the incoming alerts $\sigma \in \theta$. Once the set

of possible alert batches Θ and each alert category's respective work-times $\mathcal{W}$ are known, we can construct the set of all possible analyst states, denoted $\hat{R}$. In general, $|\hat{R}|$ is bounded above by the following:

$$|\hat{R}| \leq (w_h + 1)^n \tag{1}$$

- **The Attacker:** $\mathcal{A}$ has an attack budget $B \in \mathbb{N}$ and decides when to attack and in what category. We assume $\mathcal{A}$ knows the alert level that would be generated due to their attack. The set $\mathcal{C} = \{c_h,\ c_m,\ c_l\}$ defines the respective cost to the adversary given the alert level their attack generates, where $c_h > c_m > c_l$ and $c_h, c_m, c_l \in \mathbb{N}$. $\mathcal{A}$ can attack in as many alerts as they wish as long as the sum of their costs is affordable given their current budget. The attacker's budget enables us to model the amount of risk they are willing to undertake. Attacking more frequently with attacks that generate high level alerts would likely expose $\mathcal{A}$. To capture this behavior, in the finite horizon variant, $\mathcal{A}$ starts with a full budget in the first time step and once it falls below c_l, $\mathcal{A}$ can no longer attack and the game can be considered finished. In the infinite horizon domain, if $\mathcal{A}$ chooses to abstain from attacking in a state s with budget B_s, $\mathcal{A}$ will be credited with some unit of budget in the subsequent state.
- **State Representation:** At the beginning of a time step, we assume that the system state is known to both players. Our state space is defined as follows:

$$s = [R_s | B_s] \tag{2}$$

Given the state s and alert arrival θ, both the action space of the defender $\mathcal{D}_a(s, \theta)$ and the action space of the attacker $\mathcal{A}_a(s, \theta)$ can be generated. The respective sizes of these action spaces are given in Equations 3 and 4. $\mathcal{D}_a(s, \theta)$, in Equation 3, describes all possible combinations of assigning/not assigning the incoming alerts $\sigma \in \theta$ to the available analysts. Since we consider all analysts to be homogeneous we do not need to consider which particular analyst an alert is assigned to, but only whether or not the alert is assigned. Equation 4 enumerates all the ways $\mathcal{A}$ could hijack the alerts in θ based on the current budget in s as captured by the indicator function $\mathbb{1}_{\{.\}}$ (while also allowing them to abstain from attacking altogether)

$$|\mathcal{D}_a(s, \theta)| = \sum_{i=0}^{\mathcal{F}(R_s)} \binom{|\theta|}{i} \tag{3}$$

$$|\mathcal{A}_a(s, \theta)| = \sum_{i=1}^{2^{|\theta|}} \mathbb{1}_{\{B_s \geq Bin(i-1).\mathcal{C}_\theta\}} \tag{4}$$

where $Bin(i)$ is the binary representation of i and $\mathcal{C}_\theta$ is the cost vector for the alerts in θ according to $\mathcal{W}$. For example, an alert arrival $\theta = \{\sigma_1^m,\ \sigma_2^l,\ \sigma_3^l\}$ yields a cost vector $\mathcal{C}_\theta = [3\ 1\ 1]$. Thus, given an attacker budget $B_s = 1$, $\mathcal{A}_a = \{[0\ 0\ 0],\ [0\ 1\ 0],\ [0\ 0\ 1]\}$.

Based on the alert arrival and system state we can formulate a zero-sum game as described below for both the infinite and finite horizon variants.

A. Infinite Horizon Markov Game Formulation

To formulate the Markov game in the infinite horizon domain, we consider every state s along with every batch of alerts θ. Every state-arrival pair constitutes a sub-game where the defender attempts to detect an attack through an assignment of alerts and the attacker chooses an attack that would result in an alert at a chosen category. This game can be represented as a matrix M_s of instantaneous rewards where $M_s(a, d)$ denotes the utility awarded if $\mathcal{A}$ chooses action $a \in \mathcal{A}_a(s, \theta)$ and $\mathcal{D}$ chooses action $d \in \mathcal{D}_a(s, \theta)$. Players in a Markov game follow a policy π (i.e. a probability distribution over available actions in each state) where $\pi(s, d)$ is the probability of taking action $d \in \mathcal{D}_a(s, \theta)$. Given the current state s and alert batch θ, we can formulate a zero-sum game over the payoff matrix M_s and solve it with the following linear program:

$$\max_{\pi(s,d) \in \Pi(\mathcal{D}_a)} u \tag{5}$$

$$u \leq \sum_{a \in \mathcal{A}_a(s,\theta)} \pi(s, d) M_s(a, d) \tag{6}$$

$$\sum_{d \in \mathcal{D}_a(s,\theta)} \pi(s, d) = 1 \tag{7}$$

$$0 \leq \pi(s, d) \leq 1 \qquad \forall d \in \mathcal{D}_a(s, \theta) \tag{8}$$

Objective 5 is the value of the game, constraint 6 guarantees the worst-case over the actions available to the attacker and constraints 7 and 8 ensure that we have a valid probability distribution. The solution to this linear program yields some mixed strategy $\pi(s, \cdot)$ which when sampled from produces an expected value of playing the game in that state. Once players have chosen their actions they are awarded some utility according to M_s and the system evolves into some new state according to the transition function, wherein the process repeats.

B. Infinite Horizon Solution

To solve a given Markov game, we seek to obtain some optimal policy π^* which will maximize our worst-case expected utility over an infinite horizon. To do this we must first notice that every state's payoff matrix is incomplete, in that it only reflects the immediate rewards each action pair may lead to. If we wish to find π^*, we need every state's expected value to not just include immediate rewards but also the future rewards available from that state. This is accomplished by replacing each action pair utility in the state's payoff matrix with the quality function $Q(s, a, d)$. This quality function includes the immediate reward $M_s(a, d)$, as well as the expected discounted rewards obtainable from the future states the current action pair may lead to.

Once M_s is updated with the Q-values that reflect future rewards, we can re-solve it with our linear program and obtain

Algorithm 1 Infinite Horizon Maximin Value Iteration

Initialize:
for all s in $\mathcal{S}$ **do**
  $V_0(s) \leftarrow \max(\pi'(s,\cdot), \min(a', \text{sum}(d',\ \pi(s,d') * M_s(a',d'))))$
end for
$\delta \leftarrow 0.95$

Learn:
for $i = 1$ to iterations **do**
  for all s in $\mathcal{S}$, θ in Θ, a in $\mathcal{A}_a(s,\theta)$, and d in $\mathcal{D}_a(s,\theta)$ **do**
    $Q_i(s,a,d) \leftarrow M_s(a,d) + \delta * \text{sum}(s', \text{sum}(\theta', \Pr(\theta') * T(s,a,d,s') * V_i(s'))$
    $V_i(s) \leftarrow \max(\pi'(s,\cdot), \min(a', \text{sum}(d',\ \pi(s,\ d') * Q_i(s,\ a',\ d'))))$
  end for
end for

Algorithm 2 Finite Horizon Dynamic Programming

Initialize:
for all s in $S^{|\mathcal{T}|}$ **do**
  $V_{|\mathcal{T}|} \leftarrow \max(\pi'(s,\cdot), \min(a', \text{sum}(d',\ \pi(s,d') * M_s(a',d') + trm(s,a',d'))))$
end for

Learn:
for $t = |\mathcal{T}| - 1$ to 1 **do**
  for all s in S^t, θ in Θ, a in $\mathcal{A}_a(s,\theta)$, and d in $\mathcal{D}_a(s,\theta)$ **do**
    $Q_t(s,a,d) \leftarrow M_s(a,d) + \text{sum}(s', \text{sum}(\theta',\ \Pr(\theta') * T(s,a,d,s')\ *\ V_{t+1}(s')))$
    $V_t(s) \leftarrow \max(\pi'(s,\cdot), \min(a', \text{sum}(d',\ \pi(s,\ d') * Q_t(s,a',d')))$
  end for
end for

a new estimate for that state's value $V_i(s)$, where i corresponds to the number of iterations we have performed on the value function (e.g., $V_2(s)$ represents the value of state s considering the possible rewards an agent could accrue two steps from s).

In general, the value and quality functions are defined as follows:

$$V_0(s) = \max_{\pi \in \Pi(\mathcal{D}_a)} \min_{a \in \mathcal{A}_a} \sum_{d \in \mathcal{D}_a} \pi(s,d)\ M_s(a,d), \qquad i = 0 \tag{9}$$

$$V_i(s) = \max_{\pi \in \Pi(\mathcal{D}_a)} \min_{a \in \mathcal{A}_a} \sum_{d \in \mathcal{D}_a} \pi(s,d)\ Q_i(s,a,d), \qquad i > 0 \tag{10}$$

$$Q_i(s,a,d) = M_s(a,d) + \delta \sum_{s' \in \mathcal{S}} \sum_{\theta' \in \Theta} \Pr(\theta') T(s,a,d,s') V_{i-1}(s'), \qquad i > 0 \tag{11}$$

where δ is the discount factor such that $0 < \delta < 1$. Thus, starting with $i = 0$, and the state values initialized in Equation 9, we begin to iterate over our value function (Equations 10 and 11) where in each iteration an additional time step is included in the estimation of each state's value. The zero-sum nature of our game formulation means that an adversarial equilibrium is present in every sub-game [16]. As such, by adopting the Q-maximin value iteration methodology for Markov games, convergence of the value function is guaranteed [10]. Due to the discount factor, as the number of iterations approach infinity our value function continues to converge until we arrive at the true value function $V^*(\cdot)$. The infinite horizon algorithm to solve for the defender's optimal policy is presented in Algorithm 1.

C. Finite Horizon Markov Game Formulation

The finite horizon formulation may be more desirable than its infinite horizon counterpart in situations where a defender is aware of an imminent attack bound to occur in some time span $\mathcal{T}$ (e.g., during a military operation or a software update). In situations like this it may not be useful to maximize the utility over an infinite horizon as the defender is more concerned with immediate defenses. In this formulation, we consider a Markov game which is played over some predefined time horizon $\mathcal{T} = \{1,\ 2, \ldots,\ |\mathcal{T}|\}$. It is important to note that not every state can occur in every time step $t \in \mathcal{T}$. For example, if $B > c_h$ for all states s at time t, then it is impossible for the attacker to have a budget of zero in the next time step. As such we define time partitions on $\mathcal{S}$ such that $\mathcal{S} = \mathcal{S}^1 \cap \mathcal{S}^2 \cap \ldots \cap \mathcal{S}^{|\mathcal{T}|}$, where $\mathcal{S}^t$ denotes the set of all states occurring at time t.

D. Finite horizon solution

Similar to the infinite domain, we seek to find some optimal policy π^* which maximizes our worst-case expected utility. To do this our states' payoff matrices must not only represent the immediate payoffs available in that state, but also the expected value of the future states that current actions may lead to. The main difference in the finite domain is that a state's value not only depends on its current parameters and alert arrival batch, but also the time in which this state is occurring. To illustrate this, consider the state $s = [R_s = [5\ 5\ 5] \mid B_s = 10]$. If s occurs in the first time step the defender will be unable to allocate experts for 5 time steps, likely incurring a large loss of utility if the attacker chooses to take advantage of this. However, if this state occurs in the last time step $|\mathcal{T}|$, the defender will only be left vulnerable for one time step.

We can take advantage of this by observing that we can obtain $\pi^*(s)$ for every state $s \in \mathcal{S}^{|\mathcal{T}|}$ because these states' payoff matrices reflect the true value of the state. This is due to the fact that we know $\mathcal{S}^{|\mathcal{T}|}$ contains all the possible last states we could inhabit and therefore need not consider the value of the future states when evaluating $s \in \mathcal{S}^{|\mathcal{T}|}$. This means we can solve time partition $\mathcal{S}^{|\mathcal{T}|}$ optimally by considering each game we play in $\mathcal{S}^{|\mathcal{T}|}$ as a one-shot game. This observation serves as the starting point for our finite horizon's dynamic programming approach.

As every $s \in \mathcal{S}^{|\mathcal{T}|}$ is solved, its expected value is stored in the value function $V_{|\mathcal{T}|}(\text{s})$ (notice here that the subscript of

V denotes the time partition and not the iteration number as in the infinite domain). Once we have exhausted all states in the last time partition, we move to $\mathcal{S}^{|\mathcal{T}|-1}$ and begin solving for $V_{|\mathcal{T}|-1}(\cdot)$. In general, the value and quality functions are defined as follows:

$$V_{|\mathcal{T}|}(s) = \max_{\pi \in \Pi(\mathcal{D}_a)} \min_{a \in \mathcal{A}_a} \sum_{d \in \mathcal{D}_a} \pi_d M_s(a,d) + trm_{\mathcal{D}}(s,a,d) \quad (12)$$

$$V_t(s) = \max_{\pi \in \Pi(\mathcal{D}_a)} \min_{a \in \mathcal{A}_a} \sum_{d \in \mathcal{D}_a} \pi_d \; Q_t(s,a,d), \; 0 < t < |\mathcal{T}| \quad (13)$$

$$Q_t(s,a,d) = M_s(a,d) \; + \sum_{s' \in \mathcal{S}^{t+1}} \sum_{\theta' \in \Theta} \Pr(\theta') T(s,a,d,s') V_{t+1}(s'), \; 0 < t < |\mathcal{T}| \quad (14)$$

where $trm_{\mathcal{D}}(\cdot)$ describes the value function at the terminal state for the defender. Thus, starting with $t = |\mathcal{T}|$ (Equation 12), we then begin decrementing t in each iteration, thereby performing backward induction (Equations 13 and 14) until the value function has been computed in every time partition. Due to the finite horizon nature of the problem, we no longer need a discount factor to guarantee convergence of the Q functions and consider $V_t(\cdot)$ to be the optimal value $\forall t \in \mathcal{T}$. A complete version of the finite horizon algorithm is presented in Algorithm 2.

It is worth noting that we employ a terminal-state value function $trm(\cdot)$ that evaluates the goodness of the final state in time step $|\mathcal{T}|$ for both players. Since there is no reason for an agent to try and preserve their resources (e.g., security analysts or attack budget) in the last time window, $trm(\cdot)$ captures the additional rewards to agents who act greedily in the last time step $|\mathcal{T}|$. One particular instantiation of the terminal functions can be described by the following two equations where s' represents the terminal state:

$$trm_{\mathcal{D}}(s,a,d) = M_s(a,d) \; \frac{n - \mathcal{F}(R_{s'})}{n} \quad (15)$$

$$trm_{\mathcal{A}}(s,a,d) = M_s(a,d) \; \frac{B - B_{s'}}{B}. \quad (16)$$

where in Equation 15, the ratio of busy analysts is used to award an additional fraction of the $M_s(a,d)$ utility to the defender and in Equation 16, the ratio of exhausted budget is used to award an additional fraction of the $M_s(a,d)$ utility to the attacker.

IV. EXPERIMENTAL EVALUATION

In this section we present simulation experiments to assess the effectiveness of our game-theoretic approaches at minimizing risk in comparison to other allocation strategies. Various attacker and defender strategies are implemented in these simulations. A random strategy simply chooses from a player's action space at random with each action having an equal probability. The greedy defender strategy is modeled after a simple yet intuitive heuristic wherein the highest alert categories are always given the most attention. The greedy attacker strategy is similar in that the attacker will always attack in the highest alert category available, and will continue to attack in every time step until their budget is depleted. The linear program strategy simply solves the current state's payoff matrix with a linear program to obtain a mixed strategy and can be thought of as a completely myopic version of the dynamic programming strategy.

We start off by instantiating our model and obtaining the value of the states through the dynamic programming approaches described in the previous section (Q-maximin value iteration for the infinite horizon case and backward induction with terminal state values in the finite horizon case). Upon the conclusion of the dynamic programming we obtain the optimal worst-case allocation policies for all states and run 500 independent simulations of both the infinite and finite horizon Markov games, starting from the same initial state.

Parameter	Value
Number of experts $\|R\|$	5
Attack budget B	20
Discount factor δ	0.99
Utilities $\mathcal{U}$	$u_h = 100$, $u_m = 20$, $u_l = 5$
Attack cost $\mathcal{C}$	$c_h = 8$, $c_m = 4$, $c_l = 2$
Work times $\mathcal{W}$	$w_h = 6$, $w_m = 3$, $w_l = 1$
Alert batches Θ	θ_1, θ_2, θ_3, θ_4, θ_5, θ_6
Alerts in θ_1	σ_1^m, σ_2^m, σ_3^l, σ_4^l
Alerts in θ_2	σ_1^h, σ_2^m, σ_3^m, σ_4^l, σ_5^l
Alerts in θ_3	σ_1^m, σ_2^m, σ_3^m, σ_4^l, σ_5^l, σ_6^l
Alerts in θ_4	σ_1^h, σ_2^m, σ_3^l, σ_4^l, σ_5^l, σ_6^l
Alerts in θ_5	σ_1^h, σ_2^h, σ_3^m, σ_4^m, σ_5^l, σ_6^l, σ_7^l
Alerts in θ_6	σ_1^h, σ_2^h, σ_3^h, σ_4^m, σ_5^m, σ_6^m σ_7^l, σ_8^l, σ_9^l
Arrival prob. $\Pi(\Theta)$	$\theta_1 = 0.15$, $\theta_2 = 0.21$, $\theta_3 = 0.21$ $\theta_4 = 0.18$, $\theta_5 = 0.20$, $\theta_6 = 0.05$

TABLE I: Infinite horizon parameters.

A. *Infinite Horizon Results*

In the infinite horizon case, we instantiate our model using the parameters shown in Table I – which yield a game space of 58,212 state-arrival pairs. Here, the attacker will regenerate one unit of their budget if they choose to abstain from attacking (i.e., $B_{s'} = B_s + 1$ where s' is the successor state of s).

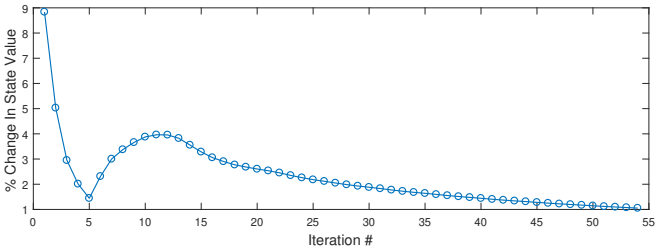

Fig. 1: Convergence of the change in the Defender's value function within the infinite horizon domain.

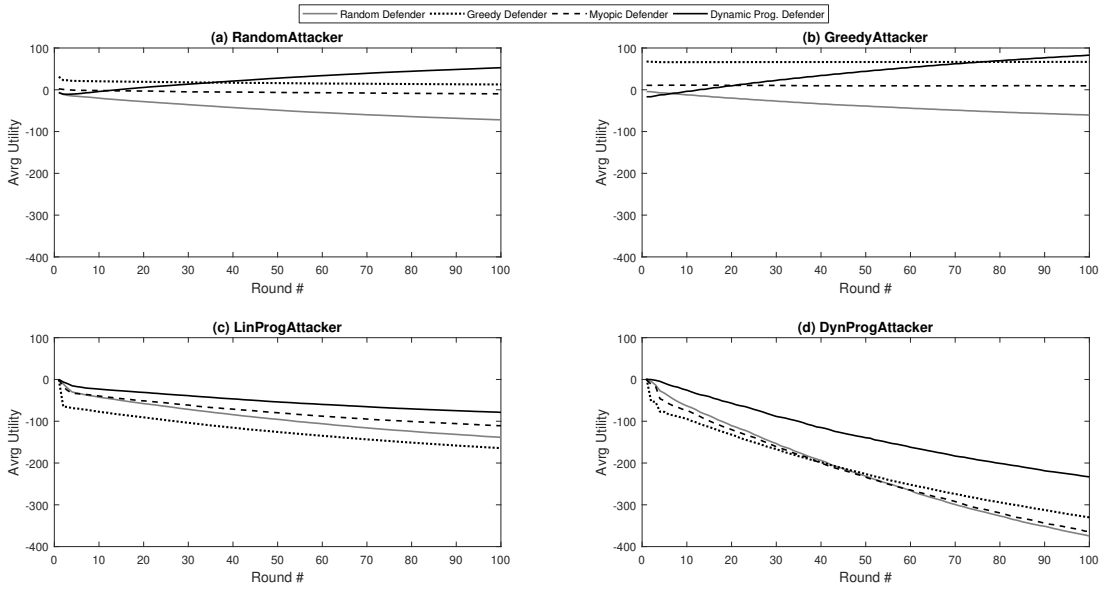

Fig. 2: Average cumulative utility accrued by the defender over 500 runs of the infinite horizon case.

Figure 1 shows the convergence of our dynamic programming algorithm's value function. The average percent change in state value is plotted on the y-axis with respect to the iteration number on the x-axis. Due to the zero-sum nature of the game, each player's value function is simply the additive inverse of the other, thus only the Defender's has been provided.

Figure 2 compares the four defender strategies against each attacker strategy in terms of the average cumulative utility over the 500 simulations – each for 100 rounds. Table II summarizes the end-game utilities where the rows correspond to the defender strategies and the columns to the attacker strategies. We underline the worst-case utility obtainable under each defender strategy.

$\mathcal{D}$ \ $\mathcal{A}$	D	L	G	R
D	<u>-223.13</u>	-78.8	82.94	52.77
L	<u>-364.27</u>	-110.78	9.45	-9.61
G	<u>-330.03</u>	-164.4	66.74	12.76
R	<u>-374.58</u>	-138.6	-60.53	-71.9

TABLE II: End-game utility in the infinite horizon case. D: Dynamic programming, L: Linear Program (myopic), G: Greedy and R: Random.

Comparing the worst-case utility amongst every defender strategy (i.e. the underlined values of Table II), we can see that the dynamic programming approach has the highest worst-case utility. This is a product of the game theoretic formulation wherein the presence of the max-min in our value functions (Equations 9 - 11) leads to finding a policy that maximizes a player's utility given that their opponent will seek to minimize that utility (recall here that the games are zero sum, so maximizing one's own utility equates to minimizing the opponent's).

As previously mentioned in [8], alert numbers typically always vastly outnumber defensive resources. As such the problem becomes one of damage control rather than damage prevention. So while a greedy defender may incur early gains against a greedy attacker (Figure 2 (b)), their over-allocation of resources to higher alert levels leads to steep loses against rational attackers who may utilize less serious vulnerabilities to orchestrate attacks (Figure 2 (d)).

Parameter	Value
Time horizon $\lvert\mathcal{T}\rvert$	10
Number of experts $\lvert R\rvert$	5
Attack budget B	24
Utilities $\mathcal{U}$	$u_h = 100$, $u_m = 20$, $u_l = 5$
Attack cost $\mathcal{C}$	$c_h = 8$, $c_m = 4$, $c_l = 2$
Work times $\mathcal{W}$	$w_h = 6$, $w_m = 3$, $w_l = 1$
Alert batches Θ	θ_1, θ_2, θ_3, θ_4, θ_5, θ_6
Alerts in θ_1	σ_1^m, σ_2^m, σ_3^l, σ_4^l
Alerts in θ_2	σ_1^h, σ_2^m, σ_3^m, σ_4^l, σ_5^l
Alerts in θ_3	σ_1^m, σ_2^m, σ_3^m, σ_4^l, σ_5^l, σ_6^l
Alerts in θ_4	σ_1^h, σ_2^m, σ_3^l, σ_4^l, σ_5^l, σ_6^l
Alerts in θ_5	σ_1^h, σ_2^h, σ_3^m, σ_4^m, σ_5^l, σ_6^l, σ_7^l
Alerts in θ_6	σ_1^h, σ_2^h, σ_3^h, σ_4^m, σ_5^m, σ_6^m σ_7^l, σ_8^l, σ_9^l
Arrival prob. $\Pi(\Theta)$	$\theta_1 = 0.15,\ \theta_2 = 0.21,\ \theta_3 = 0.21$ $\theta_4 = 0.18,\ \theta_5 = 0.20,\ \theta_6 = 0.05$

TABLE III: Finite horizon parameters.

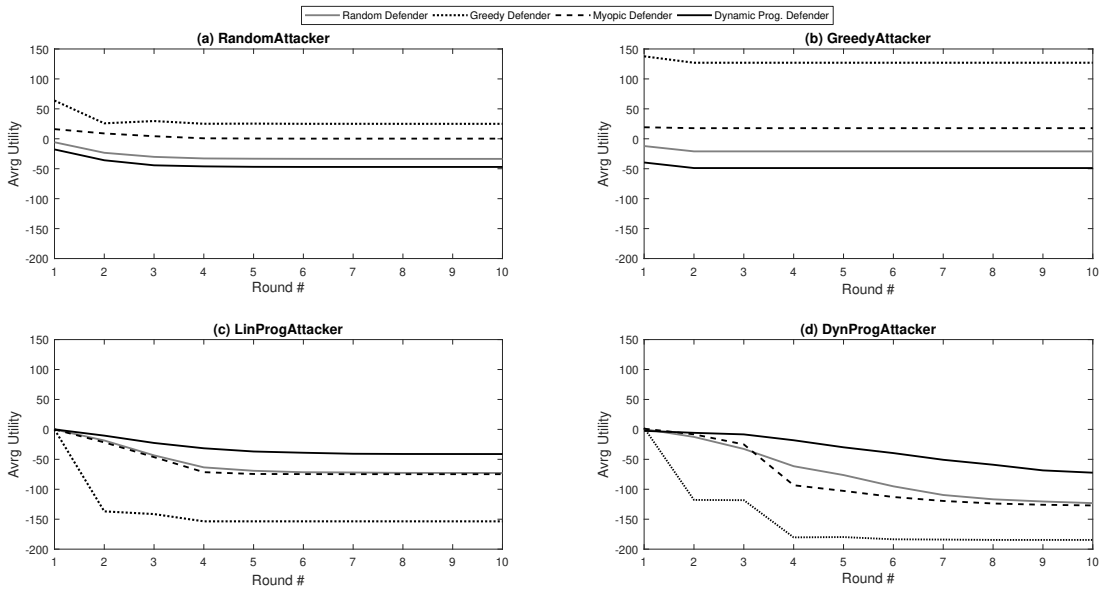

Fig. 3: Average cumulative utility accrued by the defender in finite horizon case.

B. Finite Horizon Results

In the finite horizon case, we instantiate our model using the parameters shown in table III – yielding a game space of 665,280 state-arrival pairs.

Figure 3 compares the four defender strategies against each attacker strategy in terms of the average cumulative utility obtained over the 500 simulation runs. These utilities can be seen to remain constant once the attacker has depleted their budget.

Table IV summarizes the end-game utilities where the rows correspond to the defender strategies and the columns to the attacker strategies. Once again, we can see that the dynamic programming strategy has the highest worst-case utility among all defender strategies.

$\mathcal{D}$ \ $\mathcal{A}$	D	L	G	R
D	<u>-72.30</u>	-41.17	-48.87	-47.03
L	<u>-127.02</u>	-74.77	17.74	0.28
G	<u>-184.57</u>	-153.57	127.01	24.98
R	<u>-123.14</u>	-72.77	-21.03	-33.57

TABLE IV: End-game utility in the finite horizon case. D: Dynamic programming, L: Linear Program (myopic), G: Greedy and R: Random.

Compared to the other strategies our dynamic program performs best when facing a rational opponent (i.e. an opponent who attempts to optimize the expected utility via some mixed strategy). While the greedy defender strategy may out-perform it when the attacker is irrational (Figure 3 (a) and Figure 3 (b)), it suffers a large loss of utility when the Attacker does play rationally (Figure 2 (c) and 2 (d)). This exemplifies the usefulness of the game-theoretic formulation which finds the optimal worst-case policy. This policy guarantees that an agent will value safe and consistent strategies over more bold ones which may perform well against some opponents while accruing high losses against others.

V. CONCLUSIONS

In this paper we address the resource allocation problem faced by cyber-security teams attempting to screen incoming alerts for possible attacks. We present a Markov game formulation of the problem in both the finite and infinite horizon domains and evaluate the viability of various defender strategies therein. Our formulation considers a *stateful* representation of the resources available to the players and captures the *uncertainty* in the alert arrival process. A maximin value iteration algorithm is defined and is shown to converge to a policy guaranteeing worst-case optimality. The Markov game formulation presented in this paper offers a sensible definition of optimal behavior within the cyber-security resource allocation domain and results in a policy that successfully minimizes risk when faced with a large degree of uncertainty over future states. Overall this paper demonstrates the usefulness of a game-theoretic approach to security, as it is often considered good practice for cyber-security teams to operate under the expectation of the worst-case scenario (which is exactly what our maximin strategy optimizes for).

We were able to obtain optimal worst-case policies, however, how these policies should be implemented in a real-life setting requires some additional work. In particular, often times the derived policy requires the defender to keep some resources unassigned in a time slot – so as to be ready in the

case of a high volume (or high severity) of alerts arriving in the next time slot. In practice, all analysts should be assigned at all times to ensure the widest coverage possible. One solution could be for an organization to view these types of game theoretic policies as security thresholds, wherein as long as the specified number of alerts are assigned, worst-case optimality can be maintained. Any temporary assignment of otherwise "free" analysts could only be beneficial as long as they could be subjected to reassignment in the next time slot if necessary. Such mappings of game theoretic solutions to real-world settings could be an interesting subject for further research. Furthermore, as we consider scenarios with larger numbers of resources, alerts, and actions, the curse of dimensionality begins to make obtaining optimal policies intractable. As such, one of our ongoing works is developing approximation techniques to obtain potent game theoretic policies for larger scale problems.

ACKNOWLEDGMENTS

This research was supported in-part by NSF CNS award #1149397 and was performed in-part under an appointment to the U.S. Department of Homeland Security (DHS) Science & Technology (S&T) Directorate Office of University Programs Summer Research Team Program for Minority Serving Institutions, administered by the Oak Ridge Institute for Science and Education (ORISE) through an interagency agreement between the U.S. Department of Energy (DOE) and DHS. ORISE is managed by ORAU under DOE contract number DE-SC0014664. All opinions expressed in this paper are the authors and do not necessarily reflect the policies and views of DHS, DOE or ORAU/ORISE.

REFERENCES

[1] D. Altner and L. Servi. A two-stage stochastic shift scheduling model for cybersecurity workforce optimization with on call options. 2016.

[2] T. Brook and M. Winter. Hackers Penetrated Pentagon Email, 2015.

[3] M. Brown, A. Sinha, A. Schlenker, and M. Tambe. One Size Does Not Fit All: A Game-Theoretic Approach for Dynamically and Effectively Screening for Threats. In *AAAI conference on Artificial Intelligence*, 2016.

[4] P. Elkind. Inside the Hack of the Century.

[5] J. Finkle. U.S. official sees more cyber attacks on industrial control systems, 2016.

[6] R. Ganesan, S. Jajodia, S. A., and H. Cam. Optimal Scheduling of Cybersecurity Analysts for Minimizing Risk. *ACM Trans. on Intelligent Systems and Technology*, 2015.

[7] R. Ganesan, S. Jajodia, S. A., and H. Cam. Dynamic Scheduling of Cybersecurity Analysts for Minimizing Risk Using Reinforcement Learning. *ACM Trans. on Intelligent Systems and Technology*, 2016.

[8] P. Institute. The Cost of Malware Containment, 2015.

[9] M. Jain, E. Kardes, C. Kiekintveld, F. Ordónez, and M. Tambe. Security games with arbitrary schedules: A branch and price approach. In *Proceedings of AAAI*, 2010.

[10] M. Littman. Value-function reinforcement learning in markov games. *Princeton University Press*, 2000.

[11] A. Schlenker, H. Xu, M. Guirguis, M. Tambe, A. Sinha, C. Kiekintveld, S. Sonya, N. Dunstatter, and D. Balderas. Don't bury your head in warnings: A game-theoretic approach for intelligent allocation of cyber-security alerts. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 381–387, 2017.

[12] A. Schlenker, H. Xu, M. Guirguis, M. Tambe, A. Sinha, C. Kiekintveld, S. Sonya, N. Dunstatter, and D. Balderas. Towards a game-theoretic framework for intelligent cyber-security alert allocation. In *Proceedings of the 3rd IJCAI workshop on Algorithmic Game Theory, Melbourne, Australia*, 2017.

[13] A. Shah, R. Ganesan, S. Jajodia, and H. Cam. A methodology to measure and monitor level of operational effectiveness of a csoc. *International Journal of Information Security*, pages 1–14, 2017.

[14] X. Shu, K. Tian, A. Ciambrone, and D. Yao. Breaking the target: An analysis of target data breach and lessons learned. *CoRR*, 2017.

[15] A. Sinha, T. Nguyen, D. Kar, M. Brown, M. Tambe, and A. Jiang. From Physical Security to Cybersecurity. *Journal of Cybersecurity*, 1(1):19–35, 2015.

[16] J. Von Neumann and O. Morgenstern. Theory of games and economic behavior. 1947.

[17] Z. Yin, A. Jiang, M. Tambe, C. Kiekintveld, K. Leyton-Brown, T. Sandholm, and J. Sullivan. Trusts: Scheduling randomized patrols for fare inspection in transit systems using game theory. In *Proceedings of the 24th IAAI*, Palo Alto, CA, 2012.

[18] C. Zimmerman. Ten strategies of a world-class cybersecurity operations center. *MITRE corporate communications and public affairs*, 2014.